Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica e de Computação
Departamento de Comunicações

SISTEMA DE RECONHECIMENTO DE FALA BASEADO EM REDES NEURAIS ARTIFICIAIS

Autor: Fernando Oscar Runstein

Orientador: Prof. Dr. Fábio Violaro

Tese apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas, Unicamp, como parte dos requisitos exigidos para a obtenção do título de **Doutor em Engenharia Elétrica.**

Outubro de 1998

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA – BAE – UNICAMP

R875s

Runstein, Fernando Oscar

Sistema de reconhecimento de fala baseado em redes neurais artificiais / Fernando Oscar Runstein – Campinas, SP: [s.n.], 1998.

Orientador: Fábio Violaro.

Tese (doutorado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Reconhecimento automático da voz. 2. Reconhecimento de padrões. 3. Reconhecimento de palavras. 4. Redes neurais (Computação). I. Violaro, Fábio. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Resumo

Neste trabalho são comparadas diferentes configurações de redes neurais, diversos tipos de análise do sinal de voz e diferentes parâmetros de entrada da rede neural, com o objetivo de definir o melhor sistema de reconhecimento de fala para palavras isoladas, independente do locutor e baseado em redes neurais artificiais. Um dos problemas abordados é o das redes neurais terem um número fixo de entradas enquanto as palavras a reconhecer terem durações diferentes. Duas soluções são propostas para resolver este problema: dizimação/interpolação de quadros analisando as palavras com quadros de duração fixa e dizimação/interpolação de quadros usando análise síncrona com o *pitch*. Ambos métodos apresentaram melhores resultados que os usualmente utilizados. Também é proposto um novo método de adaptação do sistema de reconhecimento de fala às características espectrais da voz do locutor, de forma a melhorar os índices de reconhecimento do sistema. Com este método conseguiu-se diminuir as taxas de erro em até 18%. Os sistemas foram avaliados com sinais ruidosos e sem ruído. Em testes independentes do locutor realizados com vocabulários de 10 a 32 palavras, obtiveram-se taxas de acerto superiores a 96%.

Este trabalho foi parcialmente financiado pelo Centro Nacional de Pesquisas –CNPq–, pelo convênio Unicamp-Telebrás 387/90 e pela FAPESP (Fundação de Amparo a Pesquisa do Estado de São Paulo, Proj. 93/0565-2).

Abstract

In this work we compared different neural network configurations, different speech analysis procedures and different neural net input parameters. The goal was to define the best *isolated word, speaker independent, speech recognition system* based on artificial neural networks. One of the problems we worked on was how to deal with different word duration and fixed number of inputs of a neural network. Two solutions are proposed to solve this problem. One of them, pitch-synchronous analysis, is new in speech recognition and produced very good results. It is also proposed in this work, a new method to adapt the speech recognition system to the spectral characteristics of the speaker's speech, in order to improve the recognition rate. With this method we diminished the error rates up to 18%. The systems were assessed with noise and noiseless signals. On speaker independent tests with 10 to 32 word vocabularies, we obtained word recognition rates better than 96%.

This work was partially supported by CNPq *(Centro Nacional de Pesquisas)*, Unicamp-Telebrás Contract 387/90, and FAPESP (*Fundação de Amparo a Pesquisa do Estado de São Paulo*, Project 93/0565-2).

Aos meus pais María Inés e Fernando Isaac, a minha madrinha Susana e a meu filho Brunito David.

Agradecimentos

Ao meu orientador *Prof. Dr. Fábio Violaro*, pelo seu apoio, dedicação, paciência, e orientação ao longo deste trabalho. Sem dúvida foram fundamentais para que este trabalho chegasse a término.

Agradeço também aos pesquisadores *Edson José Nagle, José Antônio Martins e Cairo Humberto da Silva*, do CPqD, pelas discussões técnicas que em muito contribuíram para o desenvolvimento deste trabalho. Muitas das idéias aqui implementadas surgiram de conversações frutíferas com estas pessoas, em especial com *Edson José Nagle* a quem agradeço especialmente.

Também vai meu agradecimento aos amigos e colegas da Unicamp e do CPqD, entre eles: *Ana Eliza Faria e Silva, Antonio Claudio França Pessoa, Carlos José de Campos Matos, Edwin Maurício Loboschi, Eliane Ribeiro, Fábio de Souza Azevedo, Fernando Ituo Higashie, Fernando Tofoli Queiroz, Flávia Martino Ferreira, João Luiz Ribeiro, João Batista Rickheim Filho, José Eudôxio C. de Queiroz, Jozué Vieira Filho, Leonardo Silva Resende, Marcelo Jara, Renato Marche, Ricardo Nishihara e Vitor Ciciliato.*

Gostaria também de agradecer aos professores *Amauri Lopes, José Geraldo Meloni, José Geraldo Chiquito, José Augusto Fernandes Afonso e Yuzo Iano*. Às secretárias do convênio Unicamp-Telebrás, *Celi e Marcinha.*

Aos amigos *Adrián De Witt Batista, Álvaro de Camargo Andrade, Ángelo Rossi, Joel Camargo, Jorge Rocha, Lilian Ramos, Marcelo Olguín, Martín Graciarena, Ricardo Miazzo Cuello, Ricardo Saad e Ricardo Wehbe.*

Agradeço aos que colaboraram com seu trabalho e suas vozes para criar a base de dados.

Aos meus pais, *María Inés e Fernando Isaac*, ao meu filho *Brunito David*, a *María Haydeé, María Inés, Ricardo, Miguel, Maurício, Agustín, Cecília, Diego, Mirta del Cármen, D. Luisa e Susana.*

Finalmente o meu agradecimento a *Regina Maschio*, por seu amor e leveza.

A todas as pessoas aqui mencionadas, e às não mencionadas mas que levo no coração, o meu muito obrigado.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Sistemas de Reconhecimento de Fala

1.1 Introdução

Os sistemas de reconhecimento de fala, onde uma ou várias pessoas podem comandar com a fala uma máquina ou dispositivo e, em um estágio mais avançado e ainda não atingido, conversar com a máquina, tem sido a meta dos cientistas que pesquisam na área de reconhecimento de fala. Na atualidade, os principais centros de pesquisas do mundo possuem protótipos de laboratório que permitem reconhecer desde palavras isoladas até fala contínua, com índices de acerto elevados. Muitos deles estão comercialmente disponíveis, mas ao passar do laboratório ao ambiente real de utilização os problemas se multiplicam, com a conseqüente queda no desempenho do sistema.

Os avanços significativos que vem ocorrendo nos últimos anos na área de reconhecimento de fala, podem ser atribuídos, principalmente, a três fatores: desenvolvimento de novos algoritmos de processamento digital de sinais, barateamento e aumento de capacidade dos computadores, o que permite a simulação e teste de algoritmos de reconhecimento complexos, e pesquisa interdisciplinar na área. Como conseqüência destes avanços, a comunicação homem-máquina através da fala está cada vez mais próxima de tornar-se uma realidade.

Os sistemas capazes de reconhecer comandos falados ou, ainda, fala contínua, são chamados genericamente de **Sistemas de Reconhecimento de Fala.** Estes sistemas baseiam-se na análise dos sinais acústicos da fala, para extrair de tais sinais **parâmetros** que permitam a discriminação das diferentes palavras a reconhecer.

Um sistema de reconhecimento de fala **ideal**, seria aquele que reconhecesse a fala de **qualquer pessoa**, em **qualquer tipo de ambiente** e sem nenhuma restrição quanto ao **conteúdo**

do discurso e **estilo da pessoa falar**; e isto, ainda, em **tempo real**. Ainda se está longe de conseguir este objetivo, porém as perspectivas são promissoras.

Uma das maiores dificuldades que surge na pesquisa em reconhecimento de fala é a sua natureza eminentemente interdisciplinar. A tecnologia em reconhecimento de fala envolve pesquisas em diferentes áreas tais como: processamento de sinais, ciências da computação, teoria da informação, reconhecimento de padrões, física acústica, lingüística, etc. Claramente, é impossível que uma única pessoa possua conhecimentos suficientes em todas estas áreas e daí a necessidade de interagir com outros especialistas de forma a poder conseguir resultados práticos e satisfatórios.

O aporte que cada uma destas áreas faz pode resumir-se em:

1. **Processamento de sinais:** extrair informações relevantes do sinal de voz, de forma eficiente e robusta. Isto inclui, entre outros itens, a aquisição do sinal de voz, sua digitalização, pré-processamento, análise espectral, etc.

2. **Ciências da computação:** estudar algoritmos eficientes para implementar em software ou hardware os procedimentos utilizados em um sistema de reconhecimento de fala.

3. **Teoria da informação e codificação:** definir os procedimentos para estimar os parâmetros dos modelos estatísticos; definir métodos para detectar a presença de um padrão de fala determinado; definir algoritmos de codificação e decodificação usados para decidir qual a seqüência de palavras mais provável de ter sido falada, etc.

4. **Reconhecimento de padrões:** algoritmos para criar padrões a partir dos dados disponíveis, agrupá-los em *clusters*, comparar padrões de entrada com padrões existentes, etc.

5. **Física acústica:** entender o relacionamento existente entre o sinal físico da fala e os mecanismos fisiológicos que a produzem (aparelho fonador), e que permitem sua percepção (aparelho auditivo).

6. **Lingüística:** estudar os sons constituintes da fala sob os aspectos acústicos e articulatórios (fonética), e sob os aspectos funcionais (fonologia); estudar a sintaxe (formas válidas de combinar as palavras), a semântica (significado do que se diz) e a pragmática do discurso (significado em função do contexto). Em sistemas de reconhecimento de fala contínua, o aporte da lingüística é muito necessário.

7. **Neurofisiologia:** entender os mecanismos de ordem elevada que acontecem no sistema nervoso central do ser humano e que permitem a produção e percepção da fala.

8. **Psicologia:** analisar porque uma aplicação que use reconhecimento de fala pode ser aceita pelo homem e outra não. Definir uma interface homem-máquina adequada, prática e fácil de usar, para que o usuário de um sistema de reconhecimento de fala aceite esta tecnologia e sinta vontade de usá-la. A psicologia, como pode se ver, está relacionada ao **uso** da tecnologia de reconhecimento e não à tecnologia de reconhecimento em si.

1.2 Reconhecimento de fala – Histórico

Este histórico não pretende ser completo, apenas mencionar os principais destaques ocorridos em reconhecimento de fala a partir de 1950, ano em que começaram a publicar-se as pesquisas nesta área. As informações provêm de Rabiner & Juang (1993) e Furui (1995).

Os primeiros sistemas projetados para reconhecimento automático de fala trataram de explorar as idéias da fonética acústica. Em 1952, pesquisadores dos laboratórios Bell construíram um sistema para reconhecer os dez dígitos da língua inglesa, falados isoladamente por um único locutor. O sistema estava baseado na medida de ressonâncias espectrais na região vocálica de cada dígito. Em 1956, os laboratórios RCA nos Estados Unidos trataram de reconhecer dez sílabas diferentes embebidas em palavras. O sistema também se valia de medidas espectrais feitas principalmente durante os sons vocálicos e era dependente do locutor. Em 1959, pesquisadores do University College da Inglaterra introduziram uma novidade em seus experimentos para reconhecimento de fonemas embebidos em palavras. Eles utilizaram *informações estatísticas* sobre seqüências de fonemas permitidas em inglês, visando melhorar o reconhecimento de palavras com mais de um fonema. Foi um trabalho pioneiro no uso de modelamento estocástico de fala para auxiliar o reconhecimento. No mesmo ano, no MIT Lincoln Lab dos Estados Unidos, foi feita a primeira tentativa de reconhecer fala independente do locutor. Visava-se reconhecer vogais embebidas em *strings* (seqüências) da forma /b/-vogal-/t/.

Na década de 60, vários laboratórios de pesquisa japoneses começaram a trabalhar na área de reconhecimento de fala alcançando bons resultados no reconhecimento de dígitos e fonemas (Kyoto University, Radio Research Lab e NTT Labs).

As idéias mais importantes surgidas naquela década foram as de predição linear (*Linear Predictive Coding* ou LPC), e de *Dynamic Time Warping* (DTW). O uso da técnica LPC foi proposto por Itakura e Saito do NTT Labs, como uma forma eficiente de análise da voz, baseada em um modelo de produção da fala. O uso da técnica DTW foi proposto por Vintsyuk (Ucrânia, ex. União Soviética), como um método para calcular a similaridade entre duas seqüências temporais (no caso, sentenças faladas). Isto permitia lidar com o problema de expansão e compressão temporal da fala, facilitando o alinhamento temporal entre sentenças. O trabalho de Vintsyuk baseava-se em programação dinâmica e ficou desconhecido no ocidente até a década de 80.

Na década de 70 os destaques foram: uso das técnicas LPC em reconhecimento de fala (até então só tinham sido usadas para codificação); começo de estudos em reconhecimento de fala contínua na Carniege Mellon University (CMU); estudos para o reconhecimento de grandes vocabulários na IBM; técnicas para reconhecimento de fala independente do locutor no Bell Labs e desenvolvimento de algoritmos para reconhecimento de palavras no NTT Labs do Japão, usando como unidades fonemas e sílabas.

Na década de 80 surgiram os métodos de *modelamento estatístico* da fala para reconhecimento da mesma. O mais conhecido destes métodos é o baseado nos Modelos Ocultos de Markov (*Hidden Markov Models* ou HMM). A teoria de HMM era conhecida havia vários anos, mas só foi utilizada para reconhecimento de fala a partir de meados da década de 80. Isto ocorreu devido à publicação e divulgação massiva dos seus métodos e teoria.

Na mesma década de 80, o que deu maior impulso à área de reconhecimento de fala nos Estados Unidos foi o projeto DARPA (*Defence Advanced Research Projects Agency*). Tal projeto (atualmente ARPA), financiou um grande programa de pesquisas com o objetivo de obter alta precisão no reconhecimento de fala contínua independente do locutor para um vocabulário de 1000 palavras. Varias instituições de pesquisa fizeram contribuições importantes neste projeto, como o Massachusets Institute of Technology, a Carniege Mellon University, MIT Lincoln Labs, AT&T Bell Labs, etc.

Nessa mesma época foi criado no Japão um grupo de pesquisas em reconhecimento de fala, sob a liderança de pesquisadores da NTT. O objetivo do grupo era ambicioso: a tradução automática de fala recebida via linha telefônica. Também foram realizadas pesquisas em vários lugares desse país para a obtenção de técnicas de adaptação de locutor em sistemas multilocutores.

Na década de 80 foi introduzida a tecnologia de redes neurais artificiais para reconhecimento de fala. As redes neurais artificiais eram conhecidas desde a década de 50, mas nessa época não mostraram utilidade devido a possuírem muitos problemas práticos. Na década de 80, entretanto, o desenvolvimento de algoritmos eficientes de treinamento das redes assim como uma maior compreensão da técnica, fizeram com que muitos sistemas de reconhecimento de fala fossem propostos e implementados com redes neurais artificiais.

Passando à década de 90, as pesquisas nesta década incluem o reconhecimento de fala contínua irrestrita, independente do locutor e com vocabulário ilimitado. Muitos problemas estão sendo estudados para conseguir este objetivo como, por exemplo, robustez ao ruído de fundo, distorções introduzidas pelo canal de transmissão, diferenças entre os microfones usados no treinamento e no uso normal do sistema, eco no sistema, vozes misturadas à do usuário, detecção de início e fim da fala, adaptação ao locutor, etc.

1.3 Caracterização dos sistemas de reconhecimento de fala

Um sistema de reconhecimento de fala é caracterizado pelo tamanho do vocabulário que reconhece, o estilo de fala que aceita e pela dependência ou independência do sistema com respeito ao locutor (usuário do sistema). O desempenho (taxa de acerto do sistema), depende fortemente destes fatores.

Com respeito ao tamanho do vocabulário, tem-se (Nejat Ince, 1992):

- **Sistemas de vocabulários pequenos:** os que reconhecem até 20 palavras.
- **Sistemas de vocabulários médios:** os que reconhecem entre 20 e 100 palavras.
- **Sistemas de vocabulários grandes:** os que reconhecem entre 100 e 1000 palavras.
- **Sistemas de vocabulários muito grandes:** os que reconhecem mais de 1000 palavras.

Estes números não são estritos, mas valores considerados razoáveis. Quanto maior for o vocabulário a reconhecer, maior será a complexidade do sistema de reconhecimento.

Deve-se considerar, também, que se as palavras do vocabulário a reconhecer são similares entre si, há mais chances do sistema errar. Assim, pode ocorrer que um determinado vocabulário permita taxas de reconhecimento maiores que um vocabulário menor com palavras similares entre si. Existem muitas tarefas úteis e interessantes que requerem um vocabulário pequeno (dígitos, comandos de menus, etc.). Portanto, não é mandatório que um sistema deva reconhecer um grande vocabulário para ser útil. Para aplicações como ditado de cartas, acesso a bases de dados com linguagem natural e outras, é claro que o sistema precisa reconhecer confiavelmente 30 mil ou mais palavras.

Com respeito ao tipo de fala que o sistema aceita, tem-se:

- **Sistemas de palavras isoladas:** reconhecem palavras faladas isoladamente, isto é, com uma pausa entre as mesmas. Um valor considerado razoável para esta pausa é de 200 ms como mínimo. O objetivo da pausa entre as palavras é facilitar a detecção de início e fim das mesmas, assim como permitir uma pronúncia clara evitando o efeito de coarticulação. A coarticulação provoca alteração na forma de pronunciar os sons, devido à influência dos sons vizinhos (na fala natural, o começo e fim das palavras tem a sua pronúncia alterada devido à união que se faz do final de uma palavra com o começo da seguinte).

- **Sistemas de fala contínua:** neste caso o usuário fala naturalmente, ocorrendo a concatenação do final de uma palavra com o começo da próxima. Por causa deste fenômeno, a complexidade de um sistema para reconhecer fala contínua é maior que a de um sistema para reconhecimento de palavras isoladas.

A literatura menciona às vezes os sistemas de **fala conectada** como sendo um tipo intermediário entre os sistemas de palavras isoladas e os de fala contínua, onde o locutor não deixaria pausa entre palavras mas teria que pronunciar claramente as mesmas (perda de naturalidade para falar). Na realidade o termo **fala conectada** refere-se à técnica de reconhecimento utilizada e não ao tipo de fala. Nesta técnica os modelos das palavras são concatenados entre si para reconhecer fala contínua, sem modelar, entretanto, a coarticulação existente na fala natural. A técnica tem resultados satisfatórios, unicamente, nas aplicações em que o usuário fala fluentemente uma seqüência **limitada** de palavras pertencentes a um **vocabulário altamente restrito**.

Se estas condições não se verificam, a técnica não é eficiente. Exemplos de aplicações onde esta técnica é útil são: discagem telefônica via fala, entrada dos dígitos do cartão de crédito, comandos com palavras combinadas, etc. Nestes casos, é normal que o usuário fale as palavras naturalmente, em grupos de 2, 3 ou 4 como máximo.

Quanto à dependência ou independência do locutor (usuário do sistema), tem-se:

- **Sistemas dependentes do locutor:** neste caso o sistema reconhece a fala da pessoa que treinou o sistema. Para pessoas que não treinaram o sistema, a taxas de acerto cai sensivelmente.
- **Sistemas independentes do locutor:** o sistema reconhece a fala de qualquer pessoa, com índices de acerto aceitáveis. Neste caso é necessário treinar o sistema com uma base de dados que inclua a fala de diferentes pessoas, com diferentes sotaques, idades, sexo, educação, etc. Quanto maior for a variedade de pessoas que formam parte da base de dados de treinamento do sistema, melhor poderá vir a ser o desempenho do mesmo.
- **Sistemas multilocutores:** neste caso o sistema de reconhecimento é treinado pelo grupo de pessoas que utilizará o mesmo. Para este grupo de pessoas o sistema terá um bom desempenho, sendo que para pessoas que não o treinaram o desempenho cairá. Quanto maior for o número de pessoas que treinarem o sistema, mais este se aproximará de um sistema independente do locutor.

1.4 Problemas e metodologias utilizadas em reconhecimento de fala

Antes de descrever as metodologias usuais utilizadas em reconhecimento de fala, mencionaremos brevemente a razão do reconhecimento de fala ser tão difícil. A resposta a esta pergunta não é única, mas o fator que mais provoca problemas é a **variabilidade.** Esta variabilidade tem origem em:

1) as diferentes formas de pronunciar uma palavra ou frase, tanto por um mesmo locutor como por locutores diferentes. Esta variabilidade é considerável e depende do estado de ânimo da pessoa, educação, sotaque, pressa e intenção ao falar, características do trato vocal, etc.;

2) variabilidade do transdutor (microfone a carvão, eletreto ou dinâmico, telefone normal ou celular, etc.), e variabilidade do canal de transmissão (linha telefônica, link de satélite, etc.);

3) variabilidade nos ruídos de fundo, incluindo eco, outras vozes no local (rádio, televisão, conversações alheias, etc.), eventos acústicos transitórios como ruídos da rua, portas que se fecham, queda de objetos, etc.;

4) variabilidade na produção da fala, como tosse, pigarro ou outros ruídos produzidos pelo aparelho fonador, hesitação para falar, etc.

Nem todas as fontes de variabilidade podem ser eliminadas e portanto algumas precisam ser modeladas pela técnica de reconhecimento utilizada.

Quanto às diferentes formas de atacar o problema de reconhecimento de fala por meio de uma máquina, existem, em geral, três abordagens: a abordagem fonético-acústica, a abordagem baseada em reconhecimento de padrões (incluindo métodos determinísticos e estatísticos), e a abordagem da inteligência artificial (Rabiner & Juang, 1993). Pode-se dizer que os métodos de reconhecimento de fala que utilizam redes neurais, pertencem tanto à segunda como à terceira categoria.

A abordagem fonético-acústica baseia-se na teoria do mesmo nome e consiste na detecção seqüencial de sons e classes de sons observando as características acústicas do sinal de voz e aplicando as relações conhecidas entre estas características e os símbolos fonéticos. A teoria fonético-acústica postula que existe um número finito de unidades fonéticas diferentes em cada língua, e que tais unidades caracterizam-se por propriedades que se manifestam no sinal de fala ou em seu espectro ao longo do tempo. A teoria assume que a variabilidade existente nas propriedades acústicas de uma unidade fonética devido às unidades fonéticas vizinhas (coarticulação), e aos diferentes falantes, é conhecida, e que as regras que governam esta variabilidade são implementáveis e aplicáveis. O método fonético-acústico de reconhecimento de fala consta de dois passos: no primeiro passo o sinal de voz é segmentado em regiões discretas onde as propriedades acústicas sejam representativas de um ou mais fonemas, etiquetando-se cada segmento com o fonema ou fonemas prováveis. Num segundo passo tenta-se determinar a palavra (ou seqüência de palavras) válida, a partir da informação do primeiro passo, levando-se em conta a consistência com o vocabulário de reconhecimento, a sintaxe da língua, etc. Como exemplo de propriedades acústicas dos sons usadas no método fonético-acústico, tem-se: de-

terminação de característica sonoro/não-sonoro, nasalidade, localização de formantes, taxa de cruzamentos por zero, etc.

A segunda abordagem, baseada em reconhecimento de padrões, consiste em armazenar em memória modelos ou padrões das palavras (ou sons ou frases) a reconhecer, e depois comparar a fala de entrada com os modelos armazenados. O padrão mais próximo da fala a reconhecer é determinado e se a semelhança é grande o suficiente, decide-se que corresponde ao que foi falado. Se a semelhança é insuficiente, opta-se por requerer do usuário uma nova entrada (repetir a palavra ou utilizar um meio alternativo para ingressar a informação). A similaridade entre a fala de entrada e os padrões armazenados é medida usando-se algum tipo de distância como, por exemplo, a euclidiana. Os modelos das palavras ou frases são criados através de um programa de treinamento. Estes modelos podem consistir de padrões espectrais e temporais típicos das palavras, médias destes padrões usando várias realizações de cada palavra (de um ou mais locutores), ou podem consistir de modelos estatísticos sofisticados incluindo médias espectrais e estatísticas da variância espectral durante o tempo de duração da fala.

A terceira abordagem –que utiliza técnicas de inteligência artificial–, é um híbrido que explora idéias e conceitos das abordagens anteriores. A idéia básica é reunir e incorporar em um *sistema expert* o conhecimento usado pelas pessoas para reconhecer fala. Assim, por exemplo, poderia se utilizar conhecimento lexical na segmentação e etiquetação de cada segmento da fala, conhecimento sintático e semântico na decisão da seqüência de palavras mais prováveis de terem sido faladas e conhecimento pragmático na decisão se a sentença faz sentido no contexto presente. Como exemplo da potencialidade destas fontes de conhecimento, considere os exemplos seguintes:

1. A inflação do mês de agosto foi de quatro por **vento**.
2. Carros tristes havia desportes grama.
3. O **rio** estava violento.

A primeira sentença é sintaticamente correta mas semanticamente inconsistente. Pode ser facilmente corrigida substituindo **vento** por **cento**, palavras foneticamente muito parecidas. A segunda sentença é sintaticamente incorreta e semanticamente inaceitável; deve-se rejeitar. A terceira sentença pode significar que a cidade de Rio de Janeiro estava violenta ou que um rio

(hídrico), estava convulsionado. A pragmática decidirá qual é a interpretação correta. Este último caso é de interpretação da fala reconhecida e não um problema de reconhecimento em si.

1.5 Análise do sinal de fala – Extração de parâmetros

Qualquer que seja o método utilizado para reconhecer fala, existe uma etapa que é comum a todos os métodos, que é a análise do sinal de fala de forma a extrair informações relevantes para o seu reconhecimento. Esta análise é feita, na prática, calculando-se parâmetros espectrais e/ou temporais a partir do sinal acústico de fala, em quadros de **n** ms e refazendo o cálculo a cada **p** ms. Usualmente escolhe-se $\mathbf{p} < \mathbf{n}$ de forma que exista superposição entre quadros consecutivos. Isto suaviza a transição entre os mesmos. A duração **n** do quadro deve ser suficientemente pequena para garantir a invariabilidade do trato vocal durante a produção do som (ou, equivalentemente, garantir a estacionariedade do sinal de fala). Porém, quadros muito pequenos implicam numa sobrecarga computacional desnecessária.

Um condicionante a mais surge quando se deseja calcular o *pitch*. Pitch é o correlato perceptual da freqüência de excitação do trato vocal (freqüência de vibração das cordas vocais ou F_0), sendo usual usar o termo pitch tanto para o correlato perceptual como para F_0. Usualmente o pitch varia de 80 Hz a 300 Hz, pudendo ser tão baixo como 50 Hz (barítonos) e tão alto como 500 Hz (sopranos). No caso de se necessitar calcular o pitch, o quadro de análise deve ter duração suficiente para incluir, no mínimo, dois períodos de pitch.

Considerando os condicionantes acima, normalmente escolhem-se quadros com duração de 20 a 35 ms (alguns autores admitem o uso de quadros de 50 ms ou maiores). Deve-se ressaltar, porém, que existem eventos na produção de sons que somente podem ser capturados com quadros menores que os mencionados (da ordem de 5 ms). Como exemplo de um tal evento tem-se o momento exato da ocorrência de uma plosiva.

Na prática, as amostras dos quadros de análise são ponderadas com diferentes tipos de janelas –de igual duração que o quadro–, sendo as mais utilizadas as de Hamming, Hanning, Kaiser e Blackman. É também comum pré-enfatizar o sinal de voz utilizando um fator de pré-ênfase de 0,9 a 0,99, de forma a compensar a queda de 6 dB/oitava do espectro do sinal de voz. Esta queda de 6 dB/oitava resulta da composição da queda de 12 dB/oitava no espectro

do pulso glotal e do ganho de 6 dB/oitava causado pela radiação nos lábios (O'Shaughnessy, 1987).

Como mencionado, a análise do sinal de fala é feita computando-se parâmetros espectrais e/ou temporais a partir do sinal acústico da fala, parâmetros estes que devem permitir a discriminação e classificação dos sons e/ou palavras faladas. A análise espectral pode realizar-se utilizando bancos de filtros, análise LPC, DFT, etc. Os parâmetros espectrais usualmente calculados são os LPC, de reflexão, cepstrais, mel-cepstrais e as derivadas destes parâmetros, como por exemplo delta-cepstrais e delta-mel-cepstrais. Por outro lado, a análise temporal é feita para calcular o perfil de energia do sinal, delta-energia, taxa de cruzamentos por zero, etc.

Considerando que cada quadro de análise gera um vetor de parâmetros espectrais e temporais, o sinal de fala fica representado por uma seqüência temporal de vetores. Fazendo-se a análise a cada 10 ms, obtém-se 100 vetores por segundo.

1.6 Mais alguns conceitos sobre reconhecimento de fala pelo método de reconhecimento de padrões

A figura 1.1, mostra o diagrama em blocos de um sistema de reconhecimento de fala que utiliza o método de reconhecimento de padrões.

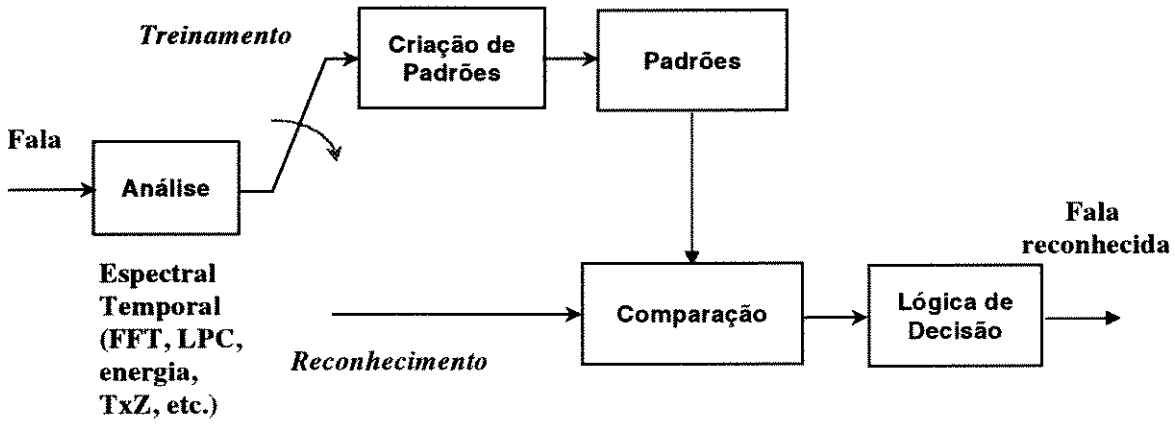


Figura 1.1: Diagrama em blocos de um sistema de reconhecimento de fala baseado em reconhecimento de padrões.

No primeiro bloco, chamado de **Análise,** o sinal de voz é analisado obtendo-se um conjunto de parâmetros representativos da palavra (ou unidade) falada. Durante a fase de treinamento do sistema (posição superior da chave), o bloco **Criação de Padrões** cria padrões (características médias dos parâmetros para uma dada palavra), ou cria modelos estatísticos (caracterizações da média e da variância dos parâmetros da fala de acordo com um modelo estatístico particular). O algoritmo para criação dos padrões é geralmente um procedimento de agrupamento das palavras em grupos consistentes ou *clusters*, de forma que a distância intracluster entre exemplos de uma mesma palavra, seja menor que a distância entre clusters de palavras diferentes. A Fig. 1.2 ilustra o resultado deste processo. Nesta figura, as realizações das diferentes palavras usadas no treinamento são representadas por pontos em um espaço bidimensional para facilitar a compreensão. No final do treinamento, cada cluster (C1 a C5) representa uma palavra diferente contendo as várias realizações da palavra em questão. Os pontos distantes de qualquer cluster (O1 a O8 no exemplo), não são levados em conta na hora de realizar a clusterização. Pode ocorrer, também, que dois ou mais clusters diferentes representem uma mesma classe (seria o caso, por exemplo, quando dois grupos de falantes pronunciam uma mesma palavra de forma tão diferente, que acabam se formando dois clusters distintos para tal palavra).

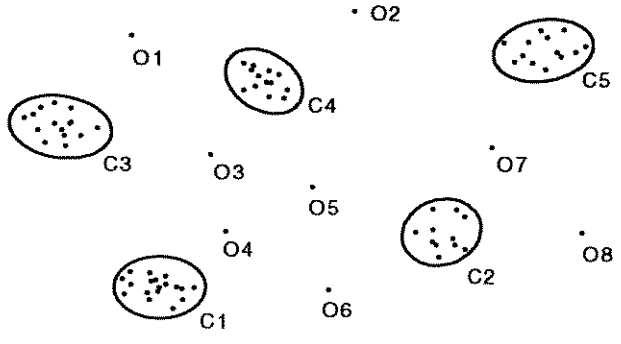


Figura 1.2: Exemplo de clusterização de palavras após treinamento do sistema.

Durante a fase de reconhecimento do sistema (Fig. 1.1, chave na posição inferior), o bloco **Comparação** tem a função de comparar o padrão desconhecido com os padrões existentes e gerar um índice de similaridade para cada comparação. Normalmente o padrão desconhecido consistirá de uma seqüência de vetores e não de um vetor único. Neste caso se faz necessário utilizar, além de uma distância local entre vetores, um procedimento de alinhamento temporal

global entre o padrão desconhecido e o padrão de referência. A razão disto é que dificilmente o número de vetores do padrão desconhecido é igual ao número de vetores dos padrões de referência. A técnica *Dynamic Time Warping* (DTW) permite –através de programação dinâmica–, determinar um caminho de alinhamento não-linear ótimo entre os padrões a comparar. A Fig. 1.3 mostra graficamente o resultado de alinhar linearmente e não linearmente – através desta técnica–, dois sinais a comparar. Pode ver-se que o alinhamento não linear resulta numa comparação mais realista da similaridade ou dissimilaridade entre os dois sinais.

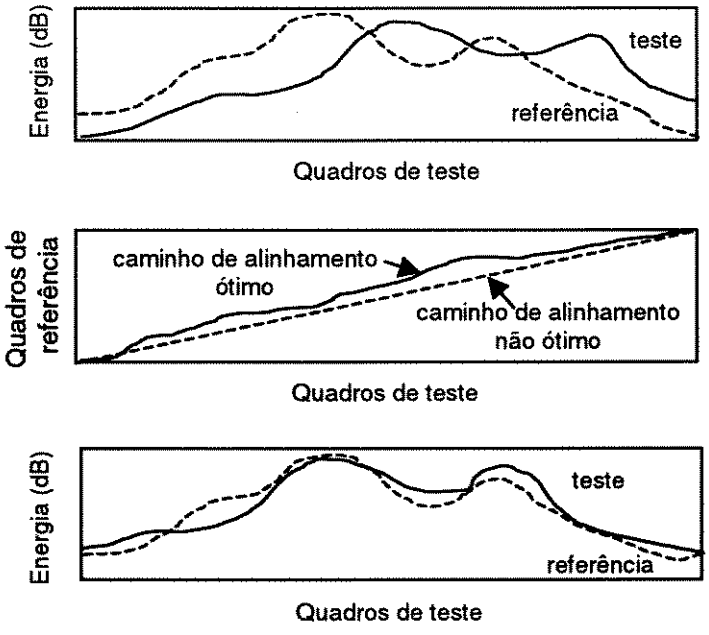


Figura 1.3: Resultado do alinhamento temporal de duas realizações de uma mesma palavra: o gráfico superior mostra o alinhamento linear; o gráfico central mostra o caminho de alinhamento ótimo e o gráfico inferior o resultado do alinhamento não linear utilizando o caminho anterior.

Uma forma de igualar o número de vetores do padrão desconhecido com o número de vetores do padrão de referência, é interpolar ou dizimar quadros (vetores) de um deles de forma a igualar o número de quadros (vetores) do outro. Perceber que isto não garante um alinhamento temporal ótimo entre os dois sinais.

O bloco final da Fig. 1.1, **Lógica de Decisão**, é o que decide qual padrão de referência está mais próximo do padrão desconhecido. Caso a similaridade entre o padrão desconhecido e os padrões de referência não atinja um limiar satisfatório, o usuário será compelido a repetir a

palavra ou frase falada, ou a entrar esta informação por algum outro meio (um teclado, por exemplo).

O método de **Reconhecimento de Padrões** é atualmente o mais utilizado para reconhecimento de fala. As razões disto são várias:

1. É relativamente simples de usar e fácil de compreender; existe uma justificativa matemática forte para muitos dos procedimentos usados no treinamento e no reconhecimento.
2. É robusto e invariante (o método, não os resultados), com respeito ao vocabulário a reconhecer, aos usuários, aos parâmetros utilizados, aos algoritmos de comparação e às regras de decisão. Isto o faz apropriado para uma ampla gama de unidades de fala (fonemas, sílabas, palavras, etc.), população de falantes, condições ambientais e/ou de transmissão, etc.
3. Os índices de reconhecimento obtidos são elevados, obtendo-se uma degradação suave na medida em que o problema de reconhecimento se torna mais complexo (vocabulários maiores ou gramática mais complexa[1]).

Quanto ao desempenho do sistema, o mesmo é sensível à quantidade de dados de treinamento disponíveis e à qualidade dos mesmos. Por qualidade, entenderemos aqui *variedade*, de forma a incluir as diferentes elocuções possíveis de uma palavra (ou som), e as diferentes condições ambientais em que se produzem (ruídos de fundo, microfones, transmissão, etc.). Em geral, quanto maior e mais variada for a base de dados de treinamento, melhor será o desempenho do sistema.

A carga computacional para treinamento e reconhecimento do sistema é linearmente proporcional ao número de padrões sendo treinados ou reconhecidos. Em conseqüência, esta carga computacional pode tornar o método proibitivo se o número de padrões ou classes de sons é muito grande. Daí que para vocabulários muito grandes se escolham como unidades de reconhecimento conjuntos pequenos de sons, com os quais seja possível gerar todas as palavras do vocabulário a reconhecer (exemplos de sons: fones, difones, trifones, sílabas, etc.).

---

[1] A complexidade ou perplexidade de uma gramática é dada, a grosso modo, pelo número médio de palavras possíveis em cada ponto de decisão. Sem uma gramática, o vocabulário inteiro deve ser considerado em cada ponto de decisão. Com uma gramática, é possível eliminar muitas palavras ou atribuir a algumas delas maiores probabilidades que a outras. Quanto maior for a perplexidade de uma gramática, maior será o número de palavras possíveis em cada ponto de decisão.

1.7 Redes neurais artificiais em reconhecimento de fala

Neste item mencionaremos brevemente como as redes neurais artificiais se inserem nos métodos de reconhecimento de fala mencionados, ou se constituem elas mesmas em um método de reconhecimento. No Capítulo 2 faremos uma exposição mais detalhada sobre redes neurais artificiais.

Nas técnicas de inteligência artificial para reconhecimento de fala, diferentes fontes de conhecimento necessitam ser definidas e estabelecidas. Dentro da inteligência artificial existem dois conceitos chaves que são: a aquisição automática de conhecimento (ou aprendizado), e a adaptação às novas condições. As redes neurais artificiais satisfazem plenamente estes dois conceitos, já que elas são capazes de aprender a partir dos exemplos que lhes são apresentados e de adaptar seu comportamento em função de novas entradas. Portanto, uma forma de implementar estes conceitos de inteligência artificial é através de redes neurais artificiais.

Nas técnicas de reconhecimento de padrões para reconhecimento de fala, o bloco **Criação de Padrões** (Fig. 1.1) implementa um algoritmo de clusterização que separa em diferentes grupos, ou clusters, as palavras/sons a reconhecer. As redes neurais de Kohonen (SOM ou *Self-Organizing Maps*), e as redes LVQ (*Learning Vector Quantization*), realizam esta clusterização automaticamente. As redes de *Kohonen-Grossberg* permitem, também, o agrupamento de diferentes clusters numa mesma classe.

As redes neurais "perceptrons multicamadas" (MLP ou *multilayer perceptron*), treinadas com o algoritmo *backpropagation*, assim como as redes TDNN (*Time Delay Neural Networks*), constituem-se em si mesmas em reconhecedoras de padrões, que podem ser utilizadas para reconhecimento de fala.

Dado que as redes neurais artificiais tem como característica inerente a **distribuição** do conhecimento ao longo de toda sua estrutura, isto as torna extremamente robustas e tolerantes a falhas. Em implementações práticas de redes neurais em hardware esta característica é altamente desejável, já que defeitos em alguns dos neurônios da rede não acarretarão prejuízos importantes ao funcionamento do dispositivo.

Outra característica que torna interessante as redes neurais é sua altíssima velocidade de resposta. Isto decorre do fato que os elementos (neurônios) que a compõem trabalham em paralelo. Como nos sistemas de reconhecimento de fala a carga computacional é elevada, esta particularidade é muito atraente.

As redes neurais são estruturas não lineares que podem realizar mapeamentos complexos entre entradas e saídas, computando as funções lineares ou não-lineares que as relacionam.

Deve-se mencionar, no entanto, que contraposta a estas vantagens está a limitação que as redes neurais possuem para lidar eficientemente com a estrutura temporal dos sinais de fala. Isto faz que seus desempenhos sejam inferiores ao das cadeias ocultas de Markov no reconhecimento de grandes vocabulários e em reconhecimento de fala contínua (cadeias ocultas de Markov é um método estatístico de reconhecimento de padrões). Esta situação poderá, no entanto, reverter-se, em função da criação de novas topologias e algoritmos de treinamento e reconhecimento com redes neurais.

1.8 Sistemas de reconhecimento híbridos

O estado da arte em reconhecimento de fala são sistemas híbridos integrando redes neurais com cadeias de Markov. Estes sistemas aproveitam as vantagens de cada um dos métodos para conseguir resultados melhores que utilizando HMMs ou redes neurais separadamente. Há diferentes aproximações para integrar estas duas técnicas (Niles and Silverman, 1990; Bourlard and Wellekens, 1990; Morgan and Bourlard, 1990 e 1995; Katagiri and Lee, 1993). Mencionaremos aqui três delas.

O primeiro tipo de integração possível é a que utiliza a rede neural como pós-processador dos HMMs. A figura seguinte, 1.4, mostra um diagrama em blocos deste tipo de sistema. Na figura, $P(O|\lambda_i)$ corresponde à probabilidade do modelo λ_i ter gerado a seqüência de observação O (a seqüência de observação é a seqüência de vetores de parâmetros espectrais e/ou temporais resultantes da análise do sinal de fala).

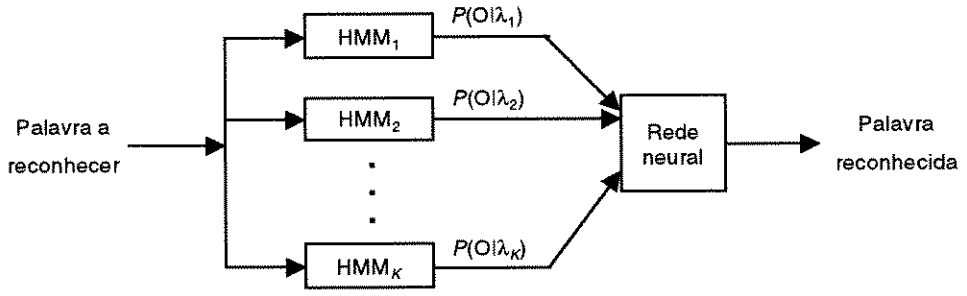


Figura 1.4: Sistema híbrido HMM-NN, utilizando a rede neural como pós-processador dos HMMs.

A idéia neste tipo de sistema é a seguinte: ao invés de escolher como palavra reconhecida aquela com maior probabilidade $P(O|\lambda_i)$, como ocorre nos sistemas HMM puros, as probabilidades de todos os modelos alimentam uma rede neural que decide qual foi a palavra falada. Perceba-se que neste caso leva-se em conta todo o conjunto de probabilidades para decidir qual foi a palavra falada, e não apenas a probabilidade de maior valor. Para a criação do HMM correspondente a cada palavra do vocabulário utilizam-se os algoritmos usuais (o *Forward-Backward* ou o algoritmo de Viterbi). No caso de um vocabulário de K palavras a reconhecer, haverá K modelos de Markov. Para o treinamento da rede neural utiliza-se o algoritmo correspondente, por exemplo, o backpropagation para a rede perceptron multicamadas (o algoritmo *backpropagation* será apresentado no Cap.2).

Um outro tipo de integração possível é a que utiliza o HMM como segmentador e normalizador temporal de um perceptron multicamadas, de forma a alimentar a rede neural com vetores de dimensão fixa. A figura seguinte ilustra este tipo de sistema para o caso de termos um vocabulário de K palavras a reconhecer.

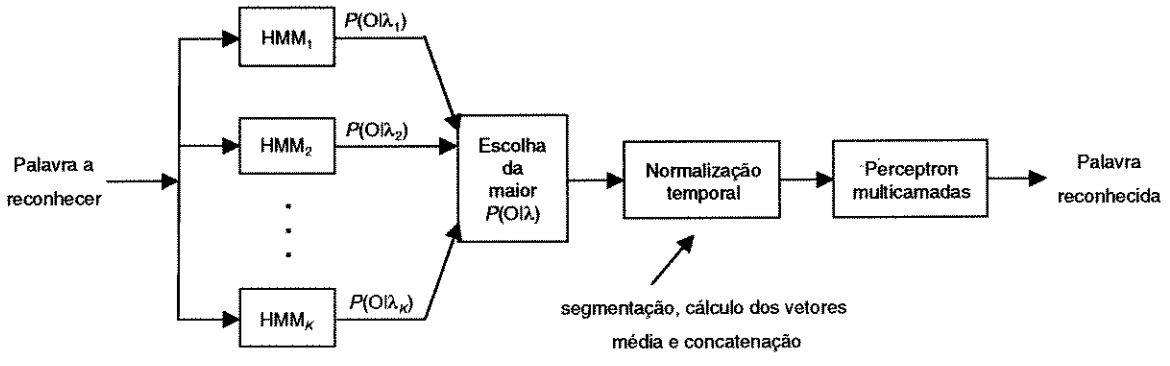


Figura 1.5: Sistema híbrido HMM-NN, que utiliza HMM como segmentador para a rede neural.

A idéia básica nesta integração é a seguinte: utilizar os modelos de Markov para segmentar e normalizar temporalmente a palavra de chegada –de duração variável–, de forma a gerar um vetor de dimensão fixa para alimentar a rede neural. Esta seria a que, em definitiva, decidiria qual foi a palavra falada. O sistema trabalha assim: uma vez treinados os modelos de Markov de forma usual (algoritmo *Forward-Backward* ou o algoritmo de Viterbi), com a restrição de que todos os modelos tenham o mesmo número de estados, a palavra de chegada é segmentada utilizando-se o modelo com maior probabilidade $P(O|\lambda_i)$. Esta segmentação consiste em achar a seqüência de estados ótima que possa ter gerado a seqüência de observação O. Uma vez achada a seqüência de estados ótima, calcula-se o vetor média de todos os vetores emitidos num mesmo estado durante a geração da seqüência de observação O (assume-se que vetores pertencentes a um mesmo estado têm características comuns). Com isto cada estado terá um vetor que o representa. Concatenando estes vetores obtém-se um vetor de dimensão fixa, independentemente do HMM utilizado (normalização temporal). Este vetor alimenta à rede neural que decide qual foi a palavra falada. Variações sobre esta idéia são possíveis (Martins, 1997).

Outro tipo de integração possível entre HMMs e redes neurais, é a que utiliza uma rede perceptron multicamadas para estimar as *probabilidades a posteriori*, que serão utilizadas pelas cadeias de Markov, para o cálculo das probabilidades de emissão de símbolos. A partir da obtenção das probabilidades de emissão de símbolos, usam-se as cadeias de Markov e seus algoritmos tradicionais para o reconhecimento da palavra falada. A Fig. 1.6 ilustra este tipo de sistema.

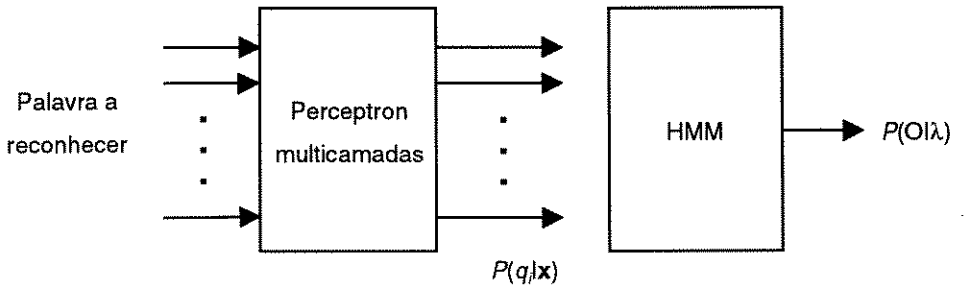


Figura 1.6: Sistema híbrido HMM-NN, utilizando a rede neural como estimador de probabilidades a posteriori para os HMMs.

Se $Q = \{q_k\}$ é um conjunto de K classes de padrões, a probabilidade a posteriori da classe q_k,, representada por $P(q_k|\mathbf{x})$, é a probabilidade de um padrão pertencer à classe q_k condicionada à ocorrência do vetor **x**.

A idéia deste sistema baseia-se na demonstração de que num perceptron multicamadas com K neurônios de saída, as saídas y_k são estimativas da distribuição de probabilidades de K classes, condicionada ao vetor **x** de entrada da rede ($y_k(\mathbf{x}) = P(q_k|\mathbf{x})$). O cálculo posterior de $P(\mathbf{x}|q_k)$, para uso nas cadeias de Markov, é feito através da regra de Bayes.

Já que estes sistemas não serão implementados, não nos estenderemos mais em sua explicação.

1.9 Avaliação dos sistemas simulados

Os sistemas de reconhecimento de fala podem ser avaliados sob diferentes aspectos, tais como: taxas de acerto e de erro do sistema, velocidade de resposta, complexidade computacional do algoritmo, possibilidade de implementá-lo em tempo real, custo, etc.

Para o cálculo das taxas de acerto e de erro, o sistema é testado com palavras que pertencem ao vocabulário a reconhecer. A equação que permite calcular a **taxa de erro,** em por cento, é a seguinte:

$$\textbf{taxa de erro}\,(\%) = \frac{\text{total de palavras reconhecidas incorretamente}}{\text{total de palavras testadas}}\ 100 \qquad (1.1)$$

De igual forma a **taxa de acerto**, em por cento, é dada por:

$$\textbf{taxa de acerto}\,(\%) = \frac{\text{total de palavras reconhecidas corretamente}}{\text{total de palavras testadas}}\ 100 \qquad (1.2)$$

Em ambos casos, o *total de palavras testadas* corresponde ao número de palavras pertencentes ao vocabulário a reconhecer, que foram submetidas ao reconhecedor. A soma da taxa de erro e da taxa de acerto deve dar 100%.

É possível submeter ao reconhecedor palavras que não pertencem ao vocabulário a reconhecer, para determinar se as rejeita adequadamente ou se as confunde com palavras válidas. Nestes casos outras medidas a calcular para avaliar a qualidade do reconhecedor são:

Taxa de rejeição (palavras corretamente rejeitadas): indica a porcentagem de palavras corretamente rejeitadas por não pertencerem ao vocabulário de reconhecimento. Corresponde ao quociente entre o total de palavras corretamente rejeitadas e o total de palavras *não pertencentes* ao vocabulário que foram submetidas ao reconhecedor.

Taxa de falsa rejeição (palavras incorretamente rejeitadas): indica a porcentagem de palavras que não deveriam ter sido rejeitadas, já que pertencem ao vocabulário a reconhecer, e que –não obstante– foram rejeitadas. Corresponde ao quociente entre o total de palavras incorretamente rejeitadas e o total de palavras *pertencentes* ao vocabulário que foram submetidas ao reconhecedor.

Taxa de falsa aceitação (palavras incorretamente aceitas): indica a porcentagem de palavras incorretamente aceitas, isto é, que deveriam ter sido rejeitadas por não pertencerem ao vocabulário a reconhecer mas que foram aceitas. Corresponde ao quociente entre o total de palavras incorretamente aceitas e o total de palavras *não pertencentes* ao vocabulário que foram submetidas ao reconhecedor.

Ao longo deste trabalho os sistemas implementados foram avaliados, unicamente, com palavras pertencentes ao vocabulário a reconhecer. Portanto as taxas calculadas para avaliá-los foram as de erro ou, equivalentemente, as de acerto (Eqs. 1.1 e 1.2).

1.10 Conteúdo da Tese

O objetivo da pesquisa realizada nesta tese, foi definir um sistema de reconhecimento de fala para palavras isoladas, independente do locutor e baseado em redes neurais. Para isto, vários aspectos foram pesquisados:

- que parâmetros utilizar como entradas da rede neural (algoritmos de obtenção dos parâmetros, complexidade computacional, índices de acerto obtidos, etc.);

- tipos de redes a utilizar (perceptrons multicamadas, redes de Kohonen-Grossberg, redes LVQ (*Learning Vector Quantization*)); número de camadas, número de neurônios, algoritmos de treinamento, etc.;
- tipos de análise do sinal de fala: análise com igual número de quadros para todas as palavras, análise com quadros de comprimento fixo, uso das técnicas *trace-segmentation*, *individual trace-segmentation*, quantização vetorial, etc.;
- dizimação e interpolação de quadros para manter constante o número de entradas da rede neural, critérios; análise com quadros de comprimento fixo e análise síncrona com o pitch;
- testes dos sistemas com e sem ruído;
- adaptação do sistema de reconhecimento de fala às características espectrais da voz do locutor.

Como se verá ao longo da tese, os melhores resultados foram obtidos utilizando perceptrons multicamadas, parâmetros mel-cepstrais obtidos via DCT (*Discrete Cosine Transform*), perfil de energia da palavra, e análise síncrona com o pitch. Este tipo de análise (síncrona com o pitch), é inédito em reconhecimento de fala. A adaptação às características espectrais da voz do locutor também introduziu ganhos significativos nos índices de acerto do sistema.

A apresentação dos itens anteriores ao longo da tese está organizada da seguinte forma:

- No Capítulo 2 é feita uma revisão de redes neurais artificiais, com ênfase nas redes utilizadas neste trabalho.
- No Capítulo 3 são descritos os parâmetros espectrais e temporais utilizados como entradas da rede neural, e a base de dados empregada para treinamento e teste dos sistemas simulados.
- No Capítulo 4 são descritos os diferentes sistemas simulados e os resultados obtidos. Nesse capítulo apresentam-se os resultados que se obtiveram com as diferentes redes neurais e com os diferentes tipos de técnicas empregadas (igual número de quadros para todas as palavras, quadros de comprimento fixo, quantização vetorial, trace segmentation, individual trace segmentation, dizimação e interpolação de quadros, análise síncrona com o pitch, comportamento dos sistemas com e sem ruído, etc.).

- No Capítulo 5 apresenta-se a técnica proposta de adaptação do sistema às características espectrais da voz do locutor e os resultados obtidos com esta técnica.
- No Capítulo 6 são apresentadas as conclusões da tese.

Capítulo 2

Redes Neurais Artificiais

As redes neurais artificiais são inspiradas nas redes neurais biológicas, i.e., são formadas por elementos que se comportam de forma similar aos neurônios biológicos naquelas funções mais elementares. Os elementos que formam uma rede neural artificial, chamados daqui em diante de *neurônios artificiais* ou simplesmente *neurônios*, estão organizados numa estrutura que pode ou não estar relacionada com a anatomia do cérebro. Apesar da muita ou pouca semelhança que possa existir entre o cérebro verdadeiro e as redes neurais artificiais, o fato é que as redes neurais artificiais exibem algumas características que fazem lembrar fortemente características cerebrais tais como: capacidade de aprender a partir de exemplos, capacidade de generalizar e capacidade de extrair características essenciais a partir de dados aparentemente irrelevantes.

Assim como as redes neurais biológicas, as redes neurais artificiais se caracterizam por:

1. Uma grande quantidade de elementos simples de processamento, os neurônios.
2. Uma rede de interconexões que ligam os neurônios entre si e que permite a troca de informações entre eles. Cada conexão tem associada um peso que pondera os sinais que circulam pela conexão e que armazenam o conhecimento da rede neural.
3. Controle paralelo distribuído.

A saída de cada neurônio depende da soma de seus sinais de entrada e de uma função de transferência aplicada a esta soma. O padrão de conexões entre os neurônios é denominado de *arquitetura* ou *topologia da rede*, o método para determinar os pesos das conexões entre os neurônios se denomina *algoritmo de treinamento ou aprendizado* e a função de transferência que determina a saída do neurônio em função de suas entradas denomina-se *função de ativação.*

As redes neurais artificiais podem ser aplicadas em uma grande variedade de problemas tais como: armazenamento e recuperação de padrões, classificação de padrões, mapeamento entre entradas e saídas de um sistema para determinar a função de transferência, agrupamento de padrões similares, implementação de funções lógicas e cálculo de soluções para problemas de otimização com restrições.

Na literatura, as redes neurais artificiais também são referenciadas como sistemas conexionistas, neurocomputadores, e processadores paralelos distribuídos.

A motivação para trabalhar com redes neurais artificiais (ou simplesmente *redes neurais*), surge da compreensão de que o cérebro se comporta de uma forma totalmente diferente à dos computadores convencionais. Enquanto em um computador convencional os circuitos operam na ordem de nanosegundos e o processamento é serial, os neurônios operam na ordem de milisegundos e o processamento é paralelo. Isto implica que os neurônios são 6 ordens de magnitude mais lentos que os circuitos eletrônicos atuais. No entanto, humanos podem executar tarefas complexas –como interpretar uma cena visual, reconhecer uma pessoa após anos sem vê-la, entender uma sentença falada, etc.–, em apenas décimos de segundo (i.e., algumas centenas de passos). Computadores não conseguem realizar algumas destas tarefas nem após horas de processamento.

Deduz-se que o cérebro consegue estas façanhas devido à enorme quantidade de elementos de processamento que possui e ao fato destes trabalharem em paralelo. Atualmente estima-se em 10^{10} o número de neurônios do córtex cerebral e entre 10^{14} e 10^{15} o número de conexões entre eles. Os pesquisadores em inteligência artificial consideram que o caminho para construir máquinas inteligentes segue nesta direção.

2.1 O neurônio artificial

O sistema nervoso humano é de uma complexidade enorme. As pesquisas em neurobiologia ainda estão longe de conseguir uma compreensão completa do funcionamento do sistema nervoso e de como ocorrem os processos. Contudo, algumas funções básicas dos neurônios tem sido determinadas e reproduzidas no neurônio artificial.

No sistema nervoso biológico cada neurônio troca sinais, em média, com outros dez mil neurônios. Na Fig. 2.1 são mostradas as partes principais de um neurônio biológico: o corpo, os dendritos e o axônio. O desenho corresponde a uma célula do tipo piramidal, uma das mais comuns no córtex cerebral. Através dos dendritos são recebidos sinais dos outros neurônios em pontos de contato chamados *sinapses*. As sinapses ponderam a intensidade dos sinais de entrada (impulsos elétricos) reforçando alguns deles e enfraquecendo outros. No corpo do neurônio estes sinais são integrados no tempo e se o nível de intensidade supera um determinado limiar, o neurônio se ativa enviando um sinal elétrico a outros neurônios através do axônio. Como no caso dos sinais de entrada, terminações sinápticas permitem a transmissão de sinais desde o axônio a outros neurônios. No caso de células piramidais, estas recebem sinais de mais de dez mil neurônios e projetam sinais a outros milhares de neurônios.

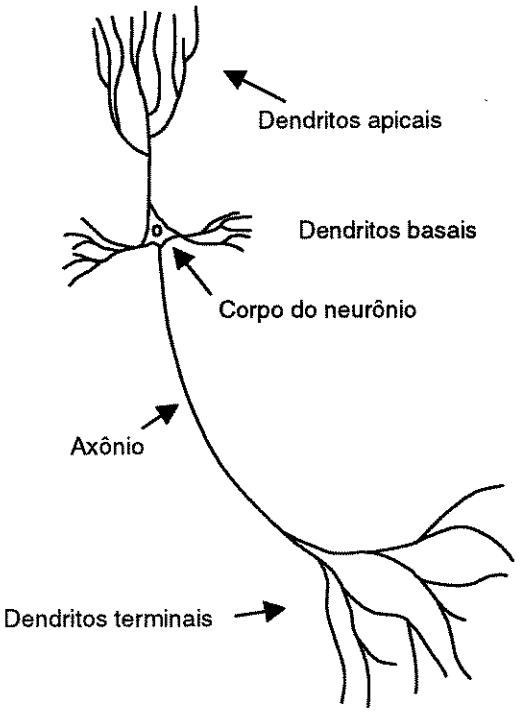


Figura 2.1: Desenho simplificado de um neurônio biológico (célula piramidal).

As sinapses são as estruturas elementares que permitem a interação entre os neurônios. O tipo mais comum de sinapse é a química, onde o sinal elétrico produzido em um neurônio é convertido em um sinal químico, transmitido a outros neurônios e convertido novamente em um sinal elétrico nos neurônios receptores. O processo químico que permite a transmissão dos

sinais entre os neurônios, também pondera o sinal correspondente (tipicamente modifica a freqüência com que os sinais são recebidos e não a intensidade). Quando no corpo do neurônio receptor acumula-se suficiente sinal em um período de tempo determinado, o neurônio se *ativa* enviando sinal para outros neurônios. Quando a sinapse é positiva, diz-se tratar-se de uma sinapse excitatória (tende a ativar o neurônio). Quando a sinapse é negativa, o efeito é contrário e trata-se de uma sinapse inibitória.

Na Fig. 2.2 é mostrado o modelo de neurônio artificial utilizado neste trabalho. Neste neurônio o processo é similar ao do neurônio biológico. Os sinais $x_1, x_2, \ldots, x_n$, provenientes de n neurônios, são ponderados pelas sinapses $w_1, w_2, \ldots, w_n$ para produzir os sinais de entrada ao neurônio y. No neurônio estes sinais são somados, produzindo o sinal y_in (Eq. 2.1). O sinal y_in é então passado por uma função de ativação $f(y\_in)$, para produzir o sinal de saída y_out (Eq. 2.2).

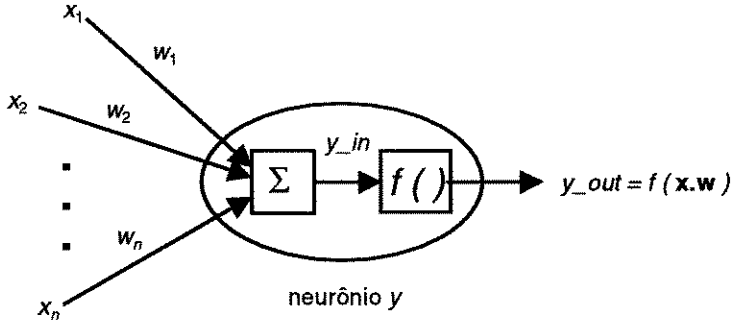


Figura 2.2: Modelo do neurônio artificial utilizado neste trabalho.

$$y\_in = x_1.w_1 + x_2.w_2 + \ldots + x_n.w_n = \sum_i x_i.w_i \,. \tag{2.1}$$

$$y\_out = f(y\_in). \tag{2.2}$$

Considerando o conjunto de entradas como um vetor **x** de dimensão n e o conjunto de sinapses como um vetor **w** também de dimensão n, pode expressar-se a saída y_in como o produto interno de ambos vetores.

$$y\_in = \mathbf{x}.\mathbf{w} \tag{2.3}$$

Usualmente inclui-se no modelo do neurônio artificial uma entrada adicional de valor 1 ou −1 (componente x_0 do vetor de entrada **x**, agora de dimensão n+1). Esta entrada é ponderada

por uma sinapse adicional w_0. Quando $x_0 = 1$, $w_0 = b_0$ denomina-se *bias* (polarizador). Quando $x_0 = -1$, $w_0 = \theta_0$ denomina-se *threshold* (limiar). O threshold tem o efeito de diminuir a amplitude da entrada à função de ativação; o bias tem o efeito de aumentá-la (isto para $w_0 > 0$).

2.2 Função de ativação dos neurônios

A função de ativação dos neurônios artificiais tem as seguintes finalidades: limitar o intervalo de valores de saída dos neurônios (razão pela qual é conhecida como *squashing function*), produzir uma não-linearidade que permita cascatear camadas de neurônios (para o qual deve ser uma função não-linear), e atuar como controle automático de ganho.

Os tipos mais usados de função de ativação são a função *degrau* e a função *sigmóide*. A função degrau é utilizada para converter entradas contínuas em saídas binárias (1 ou 0) ou bipolares (1 ou −1). Matematicamente a função degrau para saída binária (degrau binário), expressa-se da seguinte forma:

$$f(x) = \begin{cases} 1 & \text{se } x \geq \theta \\ 0 & \text{se } x < \theta \end{cases} \tag{2.4}$$

A figura seguinte mostra esta função.

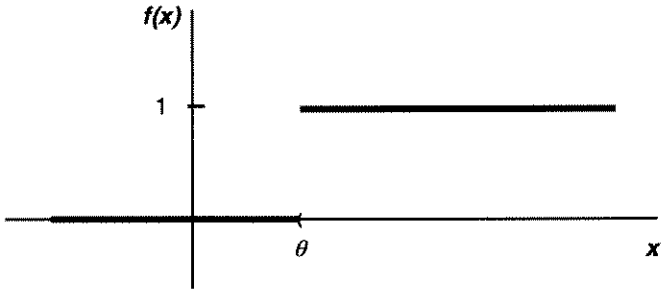


Figura 2.3: Função degrau binário.

Funções sigmóides (curvas com formato de S), são também muito usadas. As mais comuns são a função *logística* e a *tangente hiperbólica*. Elas são especialmente vantajosas em redes neurais treinadas com o algoritmo backpropagation, devido à relação simples existente entre o

valor da função em um ponto e o valor de sua derivada nesse mesmo ponto (Seção 2.4.2.1). Este fato reduz a carga computacional necessária para o treinamento.

A função *logística* (sigmóide com intervalo de saída entre 0 e 1), é freqüentemente usada como função de ativação em redes neurais em que se desejam valores de saída entre 0 e 1. As Eqs. 2.5a e 2.5b expressam matematicamente esta função e sua derivada. A Fig. 2.4 mostra a função, também chamada de *sigmóide binária*, para dois valores do parâmetro σ (pendente da sigmóide). A função logística pode ser reescalonada para que tenha uma excursão de valores mais apropriada a um dado problema. Uma excursão comum é entre −1 e 1, em cujo caso denomina-se *sigmóide bipolar*. A Fig. 2.5 mostra a sigmóide bipolar para $\sigma = 1$. As Eqs. 2.6a e 2.6b são a expressão matemática desta função e de sua derivada.

- *Sigmóide binária e sua derivada:*

$$f(x) = \frac{1}{1+\exp(-\sigma x)}, \qquad \sigma > 0 \tag{2.5a}$$

$$f'(x) = \sigma \; f(x)[1-f(x)]. \tag{2.5b}$$

- *Sigmóide bipolar e sua derivada:*

$$g(x) = 2f(x) - 1 = \frac{2}{1+\exp(-\sigma x)} - 1 = \frac{1-\exp(-\sigma x)}{1+\exp(-\sigma x)}, \qquad \sigma > 0 \tag{2.6a}$$

$$g'(x) = \frac{\sigma}{2}[1+g(x)][1-g(x)]. \tag{2.6b}$$

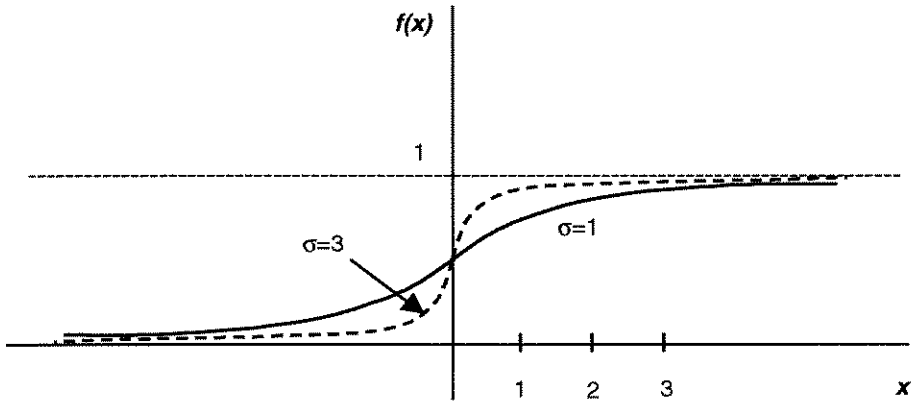


Figura 2.4: Sigmóide binária para σ = 1 e σ = 3.

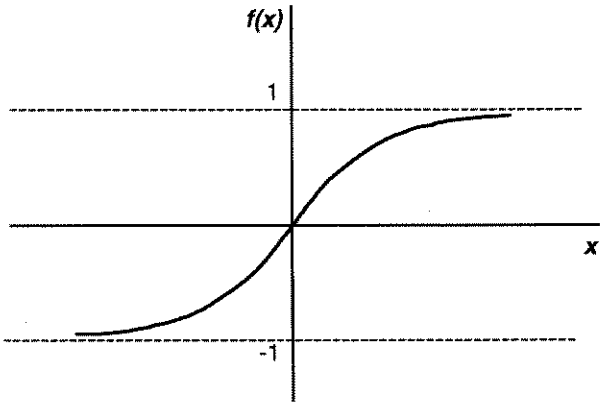


Figura 2.5: Sigmóide bipolar para σ = 1.

A *sigmóide bipolar* está intimamente relacionada com a função *tangente hiperbólica*, que também é bastante utilizada como função de ativação quando o intervalo desejado de valores de saída é entre −1 e 1. As equações seguintes, 2.7a e 2.7b, expressam matematicamente a função tangente hiperbólica e sua derivada.

- *Tangente hiperbólica e sua derivada:*

$$h(x) = \tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} = \frac{1 - \exp(-2x)}{1 + \exp(-2x)} \tag{2.7a}$$

$$h'(x) = [1 + h(x)][1 - h(x)]. \tag{2.7b}$$

Pode-se mostrar que a tangente hiperbólica equivale à sigmóide bipolar para $\sigma = 2$. Em caso de se utilizarem dados binários (0 e 1), é preferível converter os mesmos à forma bipolar (−1 e 1), e usar a sigmóide bipolar ou a tangente hiperbólica como função de ativação dos neurônios, já que isto diminui os tempos de treinamento.

Uma função de ativação que pode ser utilizada na camada de saída quando se quer interpretar as saídas da rede como probabilidades, é a *softmax*. Esta função aproxima uma sigmóide sendo sua equação:

$$f(x_i) = \frac{\exp(x_i)}{\sum_{n=1}^{m} \exp(x_n)}. \tag{2.8}$$

onde m corresponde ao número de neurônios da camada de saída. Utilizando esta função de ativação na camada de saída da rede, as saídas variam entre 0 e 1 e a soma das mesmas vale 1.

2.3 Algoritmos de treinamento

O *algoritmo de treinamento* de uma rede neural é a seqüência de passos utilizada para ajustar seus pesos. O treinamento é realizado aplicando-se seqüencialmente entradas à rede e corrigindo-se seus pesos de acordo com um procedimento determinado. Durante o treinamento os pesos da rede convergem gradualmente aos valores que fazem com que cada entrada produza a saída desejada. Existem, consensualmente, dois tipos de treinamento: *supervisionado* e *não-supervisionado*.

2.3.1 Treinamento supervisionado

Neste tipo de treinamento apresenta-se seqüencialmente à rede pares de vetores *entrada/saída-desejada*. Para cada par é calculado o erro entre o vetor de saída obtido e o vetor de saída desejado. A diferença (erro) é realimentada à rede e os pesos são corrigidos de forma a minimizar o erro. Após a apresentação completa de todos os pares de treinamento (conjunto de treinamento), tem-se completado um ciclo de treinamento denominado *época*. O processo repete-se por várias épocas até atingir-se um erro de treinamento aceitavelmente baixo.

Entre as redes que utilizam este tipo de treinamento estão os perceptrons multicamadas.

2.3.2 Treinamento não-supervisionado

Neste tipo de treinamento apresenta-se à rede uma seqüência de vetores de entrada sem especificar quais são as saídas desejadas. O algoritmo de treinamento modifica os pesos da rede de forma a produzir saídas consistentes, i.e., a aplicação de um vetor de treinamento, ou de um vetor suficientemente similar a ele, produz o mesmo padrão de saída. Neste caso o processo de treinamento extrai propriedades estatísticas do conjunto de treinamento, agrupando vetores

similares em classes ou clusters. A rede neural produz um vetor representativo para cada cluster formado. As redes auto-organizáveis inventadas por Kohonen (*Self-Organizing Maps*), assim como as ART (*Adaptive Resonance Theory*), utilizam este tipo de treinamento.

2.4 Arquiteturas de redes neurais e seus algoritmos de treinamento

2.4.1 Perceptrons

A forma mais simples de rede neural –mas que só permite classificar padrões linearmente separáveis– é o *perceptron* inventado por Rosenblatt em 1958. Rosenblatt baseou suas pesquisas em trabalhos anteriores de McCulloch e Pitts, realizados a partir de 1943. O perceptron consiste de um neurônio similar ao da Fig. 2.2, com tantas entradas quantas forem as componentes dos diferentes padrões a classificar. Sua função de ativação é um degrau binário com threshold igual a θ. Se a soma das entradas multiplicadas pelas sinapses respectivas é maior que θ, a saída é 1, caso contrário é 0. Rosenblatt provou que se os vetores (padrões), usados para treinar o perceptron provêm de duas classes linearmente separáveis, então o algoritmo de treinamento do perceptron converge e posiciona uma superfície de decisão (hiperplano), entre as duas classes mencionadas (Rosenblatt, 1962). O treshold θ possibilita que a superfície de decisão do perceptron não passe necessariamente pela origem. A prova de convergência do algoritmo de treinamento é conhecida como *Teorema de Convergência do Perceptron.* Uma vez o perceptron treinado, a apresentação de um vetor desconhecido à sua entrada produzirá uma saída de valor 0 ou 1. Associando 0 a uma classe e 1 à outra durante o treinamento, é possível classificar o vetor de entrada como pertencendo a uma destas duas classes.

Para que duas classes A e B sejam *linearmente separáveis*, é necessário que os padrões das mesmas possam ser separados por um *hiperplano*. A Fig. 2.6a mostra um exemplo de classes linearmente separáveis para o caso de duas dimensões (perceptron com duas entradas, x_1 e x_2). Na Fig. 2.6b as classes deixam de ser linearmente separáveis devido a que não é possível traçar uma reta (hiperplano de dimensão 1), que as separe.

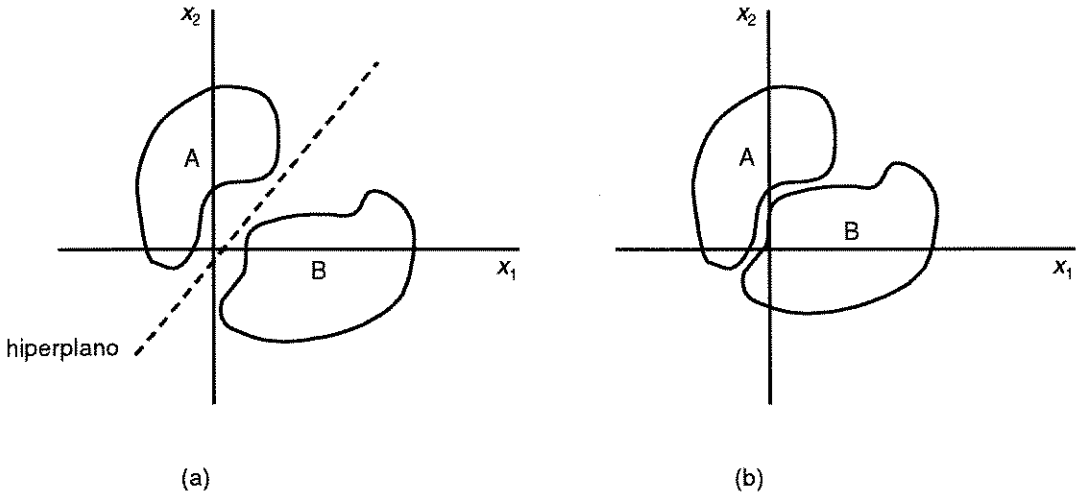


Figura 2.6: (a) Par de classes linearmente separáveis. (b) Par de classes não linearmente separáveis.

O fato de substituir a função de ativação do perceptron por uma outra função –a sigmóide, por exemplo–, não muda sua característica de poder classificar unicamente padrões linearmente separáveis (Shynk & Bershad, 1991, 1992). Pode estabelecer-se que se é utilizado um modelo de neurônio onde as entradas são combinadas linearmente e o resultado desta combinação submetido a uma função de ativação não-linear, então, independentemente da não-linearidade usada, o neurônio só consegue classificar padrões linearmente separáveis.

Quando se pretende separar padrões linearmente separáveis que pertencem a mais de duas classes, é necessário formar uma camada de perceptrons cada um representando uma das classes a separar. Mais de duas classes linearmente separáveis, significa que cada classe pode separar-se das restantes por meio de um hiperplano (ver o exemplo da Fig. 2.7a para o caso de três classes –A, B e C– no plano). A Fig. 2.7b mostra a camada de perceptrons que permite a separação das três classes da Fig. 2.7a. O neurônio *a* seria responsável pelo hiperplano *A* (neste caso uma reta), que separa a classe A das classes B e C; o neurônio *b* seria responsável pelo hiperplano *B*, que separa a classe B das classes A e C, e o neurônio *c* seria responsável pelo hiperplano *C*, que separa a classe C das classes A e B. Novamente é necessário o threshold θ para poder colocar os hiperplanos em qualquer posição e não necessariamente passando pela origem.

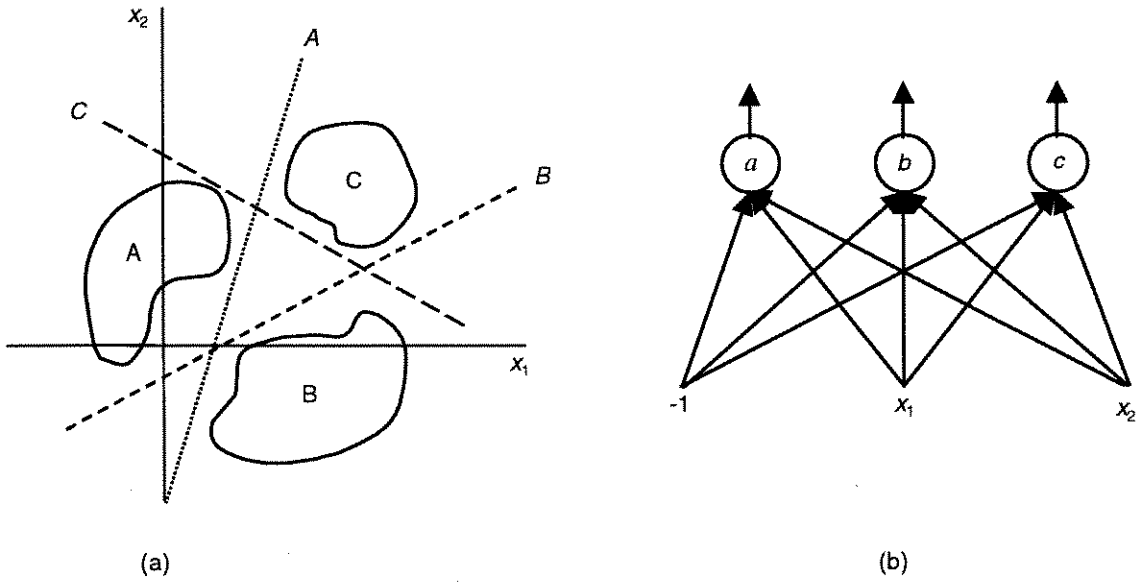


Figura 2.7: (a) Classes linearmente separáveis e hiperplanos que permitem esta separação. (b) Camada de três perceptrons usados para classificar os padrões linearmente separáveis da parte (a).

O esquema geral de uma rede neural formada por uma camada de perceptrons, é o seguinte:

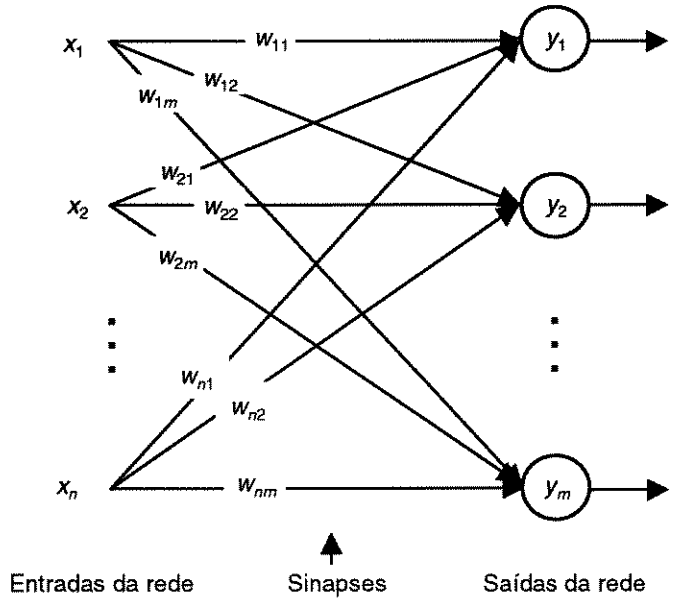


Figura 2.8: Rede neural formada por uma camada de perceptrons.

Pode ser visto que neste caso ao invés de um *vetor* de sinapses tem-se uma *matriz* de sinapses, onde o elemento w_{ij} da matriz de sinapses **W** representa a sinapse que une a entrada x_i com o neurônio de saída y_j. Em geral a matriz **W** terá dimensão n x m, onde n é o número de entradas da rede e m o número de saídas. As colunas de **W** são os vetores de sinapses dos neurôni-

os y_j. As saídas da rede podem agrupar-se em um vetor de saída **y** de dimensão *m*. O vetor **y** é obtido multiplicando-se o vetor de entradas **x** pela matriz **W**, e submetendo este produto à função de transferência dos neurônios, isto é, $\mathbf{y} = f(\mathbf{x}.\mathbf{W})$.

Para treinar a rede é necessário se dispor de vetores pertencentes a todas classes (*m* neste caso). Associando um neurônio a cada classe durante o treinamento, é possível determinar a que classe pertence um vetor desconhecido apresentado à rede.

As limitações computacionais do perceptron foram colocadas sob forma matemática por Minsky & Paper no livro *Perceptrons*, em 1969. Eles mostraram que enquanto o *Teorema de Convergência do Perceptron* garante uma classificação correta de dados linearmente separáveis, a maior parte dos problemas não possui este tipo de dados. Assim, problemas simples como a porta XOR (OU exclusivo), por exemplo, o perceptron não é capaz de resolver.

Nas próximas seções apresentar-se-ão as redes neurais e os algoritmos de treinamento utilizados neste trabalho.

2.4.2 Perceptrons multicamadas

Na seção anterior foi visto que uma camada de perceptrons (neurônios em geral) é bastante limitada em seu poder de classificação. Colocando mais de uma camada de neurônios é possível separar padrões em classes não linearmente separáveis. Em geral, uma rede neural com mais de uma camada é capaz de realizar mapeamentos complexos entre entradas e saídas. A contagem do número de camadas de uma rede neural não inclui a camada de entrada já que esta camada não realiza nenhum cálculo, servindo, apenas, para distribuir os sinais de entrada.

A Fig. 2.9 mostra uma rede neural de duas camadas: a camada *intermediária* –ou *escondida*–, e a camada de saída. Pode ser visto que agora existem duas matrizes de sinapses: a matriz **V** que une as entradas com a camada intermediária, e a matriz **W** que une a camada intermediária com a camada de saída. O número de camadas da rede é igual ao número de matrizes de sinapses.

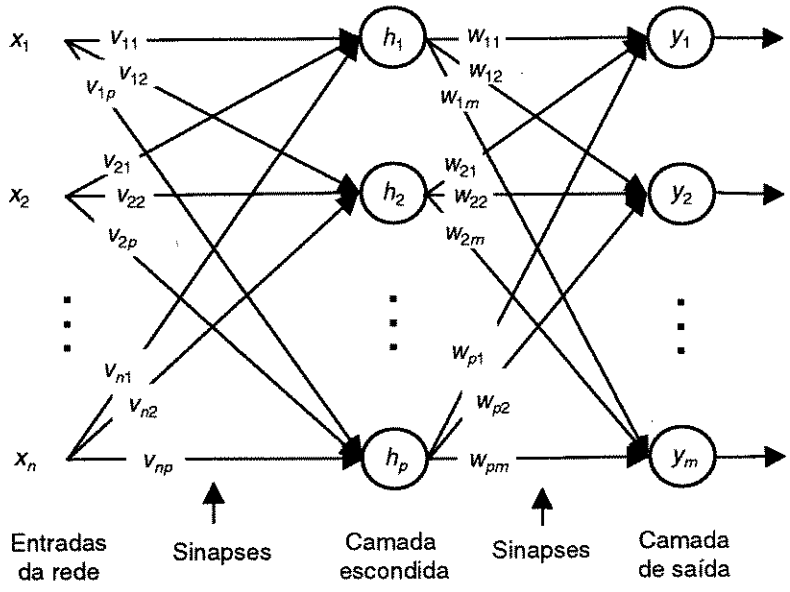


Figura 2.9: Rede neural de duas camadas (uma camada escondida).

Tipicamente, neurônios pertencentes a uma mesma camada se comportam de forma similar, i.e., possuem a mesma função de ativação e o mesmo padrão de conexões com os outros neurônios (isto não é obrigatório). Por exemplo, na Fig. 2.9 cada neurônio da camada intermediária está unido a todos os neurônios da camada de saída. Por este motivo a rede se denomina *totalmente conectada* entre camadas (todos os neurônios de uma camada estão unidos a todos os neurônios da camada seguinte). O fato de alguma sinapse ter valor 0 após o treinamento, é equivalente a que não exista a ligação entre os neurônios correspondentes.

As redes das figuras 2.8 e 2.9 são exemplos de redes *feedforward*. Neste tipo de rede os sinais fluem da entrada para a saída sempre na mesma direção (da entrada para a primeira camada intermediária, desta para a próxima camada intermediária, e assim sucessivamente até alcançar a saída). Há outros tipos de redes, como as recorrentes, onde os sinais de saída de uma camada são retroalimentados às camadas anteriores ou à própria camada.

Assim como redes neurais de uma camada permitem a separação de pontos *linearmente separáveis*, as redes neurais de duas camadas permitem a separação de pontos pertencentes a *regiões convexas* abertas ou fechadas. Uma região convexa é uma região na qual dois pontos quaisquer pertencentes à região podem ser unidos por uma linha reta que nunca abandona a região. A região é fechada se todos seus pontos estão contidos dentro de um limite (um círculo

por exemplo). É aberta se há alguns pontos fora de qualquer limite definido (a região entre duas paralelas, por exemplo). A Fig. 2.10 mostra exemplos de regiões convexas abertas e fechadas para o caso de duas dimensões (rede de duas entradas). O número de lados da região convexa é dado pelo número de neurônios da camada intermediária (cada neurônio pertencente a esta camada define uma reta que separa o plano em duas regiões.). Portanto, escolhendo-se adequadamente o número de neurônios da camada escondida é possível encerrar conjuntos arbitrários de pontos sempre que pertençam a regiões convexas. Cada neurônio da camada de saída está associado a uma região convexa diferente (isto, supondo que os pesos das sinapses correspondentes sejam diferentes).

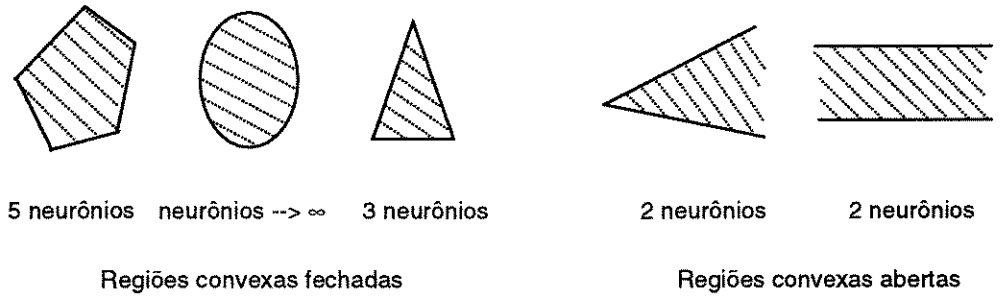


Figura 2.10: Exemplos de regiões convexas abertas e fechadas.

Uma rede neural com duas camadas escondidas (Fig. 2.11), é ainda mais geral. A sua capacidade de classificação é limitada unicamente pelo número de neurônios que possui e não há limitações de convexidade. Nestas redes a camada de saída recebe como entrada as regiões convexas formadas pela segunda camada escondida, e as combina logicamente. Esta combinação não necessita ser convexa. Como exemplo, a Fig. 2.12 mostra uma região de decisão arbitrária formada pela interseção de duas regiões convexas no plano (rede de duas entradas). Dado que se trata da interseção de dois triângulos, a primeira camada escondida deve possuir três neurônios.

Na medida em que mais neurônios são adicionados à rede, o número de lados dos polígonos convexos (dado pelo número de neurônios da primeira camada escondida), e o número de polígonos convexos (dado pelo número de neurônios da segunda camada escondida), aumenta. Isto torna possível delimitar uma região de qualquer formato com o grau de exatidão desejado. Adicionalmente, a camada de saída pode, além de intersectar regiões convexas, uni-las ou im-

plementar outras funções lógicas. Assim, conjuntos separados de pontos podem ser classificados como pertencentes a uma mesma categoria.

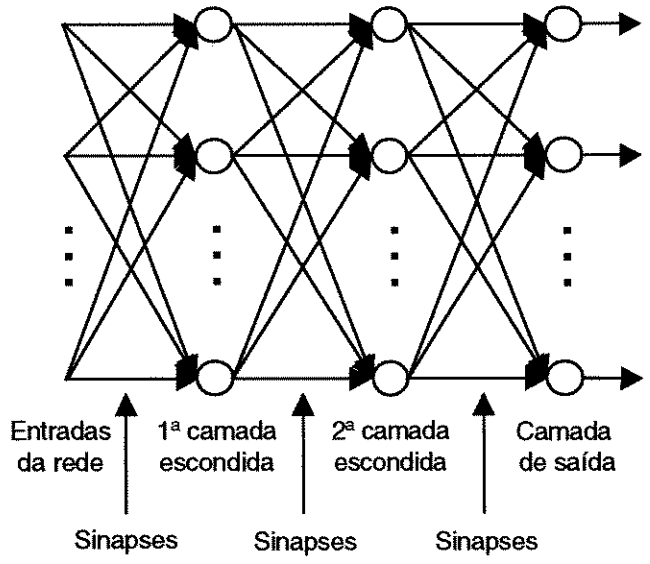


Figura 2.11: Rede neural com duas camadas escondidas (total de três camadas).

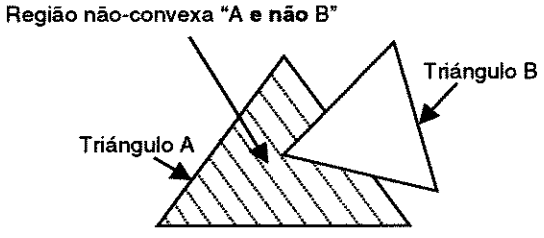


Figura 2.12: Região de decisão formada pela interseção de duas regiões convexas (função lógica: "A **e não** B").

Se as entradas da rede neural são contínuas ao invés de binárias ou bipolares, os conceitos mencionados passam a valer para regiões contínuas e não somente para conjuntos de pontos.

Não é usual o uso de redes com mais de duas camadas escondidas. No caso de mapeamento de funções contínuas, o teorema de Kolgomorov estabelece que com uma camada escondida é possível representar qualquer função (Hecht-Nielsen, 1987a).

Nas redes multicamadas é necessário que exista uma função de ativação *não-linear* entre camadas. Se isto não ocorre o poder computacional destas redes é equivalente ao das redes de

uma camada só. Considere-se, por exemplo, a rede de duas camadas da Fig. 2.9 reproduzida aqui na Fig. 2.13.

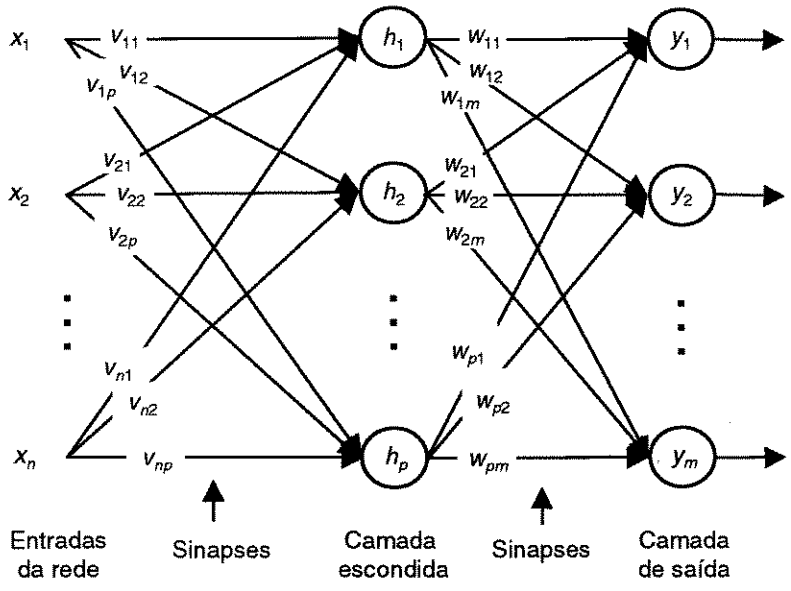


Figura 2.13: Rede neural de duas camadas (uma camada escondida).

Calcular a saída desta rede consiste em multiplicar o vetor de entrada **x** pela matriz de sinapses **V** e depois, não havendo não-linearidade entre as camadas, multiplicar o vetor resultante **h** pela matriz de sinapses **W**. A Eq. 2.9 expressa isto matematicamente.

$$\mathbf{y} = (\mathbf{x}.\mathbf{V}).\mathbf{W} = \mathbf{h}.\mathbf{W} \tag{2.9}$$

Dado que o produto de matrizes é associativo, a Eq. 2.9 pode ser substituída pela Eq. 2.10, onde a matriz **U** corresponde ao produto de matrizes **V.W**.

$$\mathbf{y} = \mathbf{x}.(\mathbf{V}.\mathbf{W}) = \mathbf{x}.\mathbf{U} \tag{2.10}$$

A Eq. 2.10 mostra que a rede original de *duas* camadas eqüivale a uma rede de *uma* camada com matriz de sinapses **U**. Portanto qualquer rede multicamadas que não possua uma função de ativação não linear entre as mesmas, pode ser substituída por uma rede equivalente com apenas uma camada. Dado que o poder computacional das redes de uma camada é limitado, é necessário usar funções de ativação não-lineares para produzir redes mais poderosas.

2.4.2.1 Algoritmo de treinamento *Backpropagation*

Como mencionado anteriormente, o algoritmo de treinamento de uma rede neural é a seqüência de passos utilizada para ajustar seus pesos. No caso de perceptrons multicamadas o algoritmo utilizado para treinamento é o backpropagation.

O algoritmo backpropagation (assim chamado porque utiliza a retropropagação do erro para corrigir os pesos da rede), teve origem em Werbos (1974), mas foram Rumelhart, Hinton e Williams que em 1986 apresentaram uma descrição clara e concisa do algoritmo. Parker também tinha publicado trabalhos a esse respeito em 1982. O algoritmo backpropagation ou *regra delta generalizada*, é simplesmente um método de gradiente descendente para minimizar o erro quadrático médio das saídas computadas pela rede, ao longo de todo o conjunto de treinamento.

A natureza geral do algoritmo de treinamento backpropagation permite que uma *rede backpropagation* (rede perceptron multicamadas, feedforward, treinada com o algoritmo backpropagation), possa ser utilizada para resolver problemas em muitas áreas diferentes. Como é o caso para a maior parte das redes neurais, o objetivo do treinamento com backpropagation é treinar a rede neural para alcançar um equilíbrio entre sua habilidade para responder corretamente a padrões de entrada que foram usados no treinamento *(memorização)*, e sua habilidade para dar uma boa resposta a entradas que são similares, mas não iguais, àquelas usadas no treinamento *(generalização)*.

O treinamento de uma rede neural usando backpropagation abrange três fases: a aplicação de um vetor de entrada e obtenção do correspondente vetor de saída, o cálculo do erro entre a saída obtida e a desejada realimentando este erro à rede, e –finalmente– o ajuste dos pesos. Após o treinamento a operação da rede consiste na aplicação de um vetor em sua entrada e na obtenção do vetor de saída. Independentemente de quão demorado possa ser o treinamento, a rede treinada pode computar sua saída muito rapidamente.

A Fig. 2.14 será utilizada para explicar o algoritmo. A figura mostra uma rede neural com uma camada escondida. A camada de entrada da rede é formada pelas entradas x_1 a x_n que distribuem seus valores à camada escondida (como explicado, a camada de entrada não se contabiliza para calcular o número de camadas da rede). A camada de entrada possui uma entrada

adicional de valor 1 que alimenta todos os neurônios da camada escondida e que atua como polarizador *(bias)* destes neurônios. A camada escondida é formada pelos neurônios h_1 a h_p, mais uma unidade adicional de valor 1 que atua como *bias* dos neurônios da camada de saída. A camada de saída é formada pelos neurônios y_1 a y_m. A camada de entrada está unida à camada escondida pela matriz de sinapses **V** e a camada escondida está unida à camada de saída pela matriz de sinapses **W**.

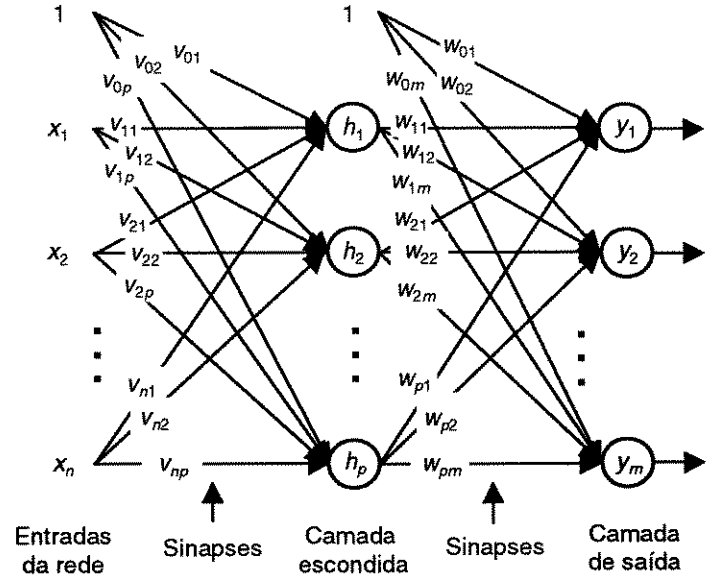


Figura 2.14: Perceptron multicamadas com uma camada escondida.

Nomenclatura a usar

A nomenclatura que utilizaremos no algoritmo backpropagation é a seguinte:

x vetor de entrada (ou de treinamento); $\mathbf{x} = (x_0, x_1, \ldots, x_n)$.

h vetor de saída da camada escondida; $\mathbf{h} = (h_0, h_1, \ldots, h_p)$.

h_in_j denota a entrada à função de ativação do j-ésimo neurônio escondido.

y vetor de saída da rede (resposta ao vetor **x**); $\mathbf{y} = (y_1, \ldots, y_m)$.

y_in_k denota a entrada à função de ativação do k-ésimo neurônio de saída.

t vetor de saída desejado *(target)*; $\mathbf{t} = (t_1, \ldots, t_m)$.

v_{ij} sinapse que une a entrada x_i com o neurônio escondido h_j.

w_{jk} sinapse que une o neurônio escondido h_j com o neurônio de saída y_k.

$\delta\mathbf{y}$ vetor delta relacionado aos erros da camada de saída; $\delta\mathbf{y} = (\delta y_1, \ldots, \delta y_m)$.

$\boldsymbol{\delta h}$ vetor delta relacionado aos erros da camada escondida; $\boldsymbol{\delta h} = (\delta h_1, \ldots, \delta h_p)$.

η taxa de aprendizagem $(0 < \eta(t) < 1)$.

α fator momento $(0 < \alpha(t) < 1)$.

$f(.)$ função de ativação dos neurônios da rede (supondo a mesma para todos eles).

Durante a fase *feedforward*, cada neurônio da camada de entrada distribui seus sinais aos neurônios da camada escondida através das sinapses v_{ij}. Os neurônios da camada escondida recebem estes sinais, computam suas saídas e as enviam aos neurônios da camada de saída através das sinapses w_{jk}. Os neurônios da camada de saída computam então suas saídas, que constituem a resposta da rede ao vetor de entrada **x**.

Na fase *backward* os neurônios da camada de saída comparam suas saídas y_k com as saídas desejadas t_k. As diferenças (erros) são determinadas e baseadas nelas calcula-se o vetor $\boldsymbol{\delta y}$. As componentes de $\boldsymbol{\delta y}$ são realimentadas à camada prévia através das próprias sinapses w_{jk}, além de serem utilizadas, posteriormente, para atualizar as sinapses w_{jk}. O vetor $\boldsymbol{\delta h}$, relacionado aos erros na camada escondida, é então computado. Não é necessário propagar as componentes de $\boldsymbol{\delta h}$ para a camada de entrada, mas estas componentes serão utilizadas para atualizar as sinapses v_{ij}. Uma vez calculados os δ's de todas as camadas da rede (menos os da camada de entrada), os pesos da rede inteira são reajustados simultaneamente. As sinapses w_{jk} são reajustadas em função de $\boldsymbol{\delta y}$ e dos h_j. As sinapses v_{ij} são reajustadas em função de $\boldsymbol{\delta h}$ e dos x_i.

O algoritmo backpropagation exige que a função de ativação $f(.)$ seja contínua, diferenciável e monotonicamente não-decrescente (Rumelhart et al., 1986). Adicionalmente é desejável que o valor de sua derivada em um ponto possa ser expressa em função do valor de $f(.)$ nesse ponto. Isto aumenta a eficiência computacional do algoritmo. A dedução do algoritmo parte da definição do erro quadrático médio das saídas da rede como:

$$E = \frac{1}{2}\sum_{k=0}^{m}(t_k - y_k)^2 ,$$

onde t_k são as saídas desejadas e y_k as obtidas. O objetivo do algoritmo é minimizar o erro quadrático médio E utilizando a regra delta generalizada. A correção nas sinapses w_{jk} que minimiza E, aplicando-se a regra delta, é dada por:

$$\Delta w_{jk} = -\eta \frac{\partial E}{\partial w_{jk}}, \qquad j = 0,..., p; \qquad k = 1,..., m. \qquad \text{(regra delta)}$$

onde η é a taxa de aprendizagem e $\partial E / \partial w_{jk}$ é o gradiente de E. O gradiente de uma função (neste caso a função é o **erro quadrático médio** das saídas da rede e as variáveis são os **pesos** da mesma), dá a direção na qual a função cresce mais rapidamente. O negativo do gradiente dá a direção na qual a função decresce mais rapidamente. Desenvolvendo a equação anterior:

$$\Delta w_{jk} = -\eta \frac{\partial E}{\partial w_{jk}} = \eta\ (t_k - y_k) \frac{\partial}{\partial w_{jk}} y_k = \eta\ (t_k - y_k) \frac{\partial}{\partial w_{jk}} f(y\_in_k)$$

$$= \eta (t_k - y_k) \frac{\partial f}{\partial y\_in_k} \frac{\partial y\_in_k}{\partial w_{jk}} = \eta\, (t_k - y_k) f'(y\_in_k) \frac{\partial y\_in_k}{\partial w_{jk}} \quad ; \; j = 0,..., p \; ; \; k = 1,..., m.$$

onde $f(.)$ é a função de ativação dos neurônios. Chamando o produto $(t_k - y_k).f'(y\_in_k)$ de δy_k e considerando que:

$$y\_in_k = \sum_{j=0}^{p} h_j w_{jk}\ , \qquad k = 1,..., m.,$$

Δw_{jk} fica (inclui-se na equação o fator tempo):

$$\Delta w_{jk}(t) = \eta\,.\,\delta y_k(t).h_j(t), \qquad j = 0,..., p; \qquad k = 1,..., m.$$

As sinapses w_{jk} são, então, corrigidas com a seguinte equação:

$$w_{jk}(t+1) = w_{jk}(t) + \Delta w_{jk}(t), \qquad j = 0,..., p; \qquad k = 1,..., m.$$

A dedução das equações que permitem corrigir as sinapses das camadas internas da rede é mais complicada e dado que não é o objetivo deste trabalho fazer tal dedução, a mesma não será apresentada.

O algoritmo backpropagation completo utilizado aqui para o treinamento dos perceptrons multicamadas, foi o seguinte:

Algoritmo backpropagation - Fase forward:

1. Inicialize os pesos da rede, a taxa de aprendizagem η e o fator momento α (quando utilizado). Cada peso deve ser inicializado com um valor randômico (aleatório) pequeno, por exemplo entre –0,1 e 0,1.

$v_{ij} = random\ (-0{,}1\ 0{,}1) \qquad \forall \quad i = 0,\ldots, n; \qquad j = 1,\ldots, p.$

$w_{jk} = random\ (-0{,}1\ 0{,}1) \qquad \forall \quad j = 0,\ldots, p; \qquad k = 1,\ldots, m.$

2. Inicialize as unidades de polarização ou bias. *Estas unidades nunca mudam seus valores.*

$x_0 = 1; \qquad h_0 = 1.$

3. Escolha um par ***entrada/saída-desejada*** *proveniente do conjunto de treinamento. Sendo* **x** *o vetor de entrada e* **t** *o vetor de saída-desejado, coloque* **x** *na entrada, propague os* x_i *à camada escondida e compute os* h_j *como:*

$$h_j = f(h\_in_j) = f\left(\sum_{i=0}^{n} x_i v_{ij}\right), \qquad j = 1,\ldots, p.$$

Envie estes sinais à camada de saída. Note que i vai de 0 *a n para incluir a unidade que atua como bias (*x_0*).*

4. Calcule os sinais de saída y_k *com a seguinte equação:*

$$y_k = f(y\_in_k) = f\left(\sum_{j=0}^{p} h_j w_{jk}\right), \qquad k = 1,\ldots, m.$$

Note que j vai de 0 *a p para incluir, também, a unidade que atua como bias (*h_0*).*

Algoritmo backpropagation - Fase backward:

5. Compute o vetor **δy** *relacionado aos erros da camada de saída. Este vetor é baseado nas saídas desejadas* t_k*, nas saídas obtidas* y_k*, e na função de ativação f(.),*

$$\delta y_k = (t_k - y_k) \cdot f'(y\_in_k), \qquad k = 1,\ldots, m.$$

Se f(.) é a sigmóide binária (Eqs. 2.5a e 2.5b), δy_k *fica:*

$$\delta y_k = (t_k - y_k) \cdot y_k \cdot (1 - y_k), \qquad k = 1,\ldots, m.$$

6. Compute o vetor **δh** *relacionado aos erros da camada escondida. Estes erros são baseados nos deltas* δy_k *da camada de saída, nas sinapses* w_{jk} *e na função de ativação f(.),*

$$\delta h_j = \sum_{k=1}^{m} \delta y_k\, w_{jk} \cdot f'(h\_in_j), \qquad j = 1,\ldots, p.$$

Corresponde, portanto, à somatória dos deltas de saída que chegam até os neurônios escondidos através das sinapses w_{jk} (retroalimentação dos erros), multiplicada pela derivada da função de ativação desses neurônios. Se f(.) é a sigmóide binária, δh_j fica:

$$\delta h_j = \sum_{k=1}^{m} \delta y_k \, w_{jk} \, . \, h_j \, (1 - h_j) \quad , \qquad j = 1, \ldots, p.$$

7. Ajuste os pesos w_{jk} entre a camada escondida e a de saída, segundo as equações seguintes (por serem as equações finais inclui-se nelas o fator tempo):

$$w_{jk}(t+1) = w_{jk}(t) + \Delta w_{jk}(t), \qquad j = 0, \ldots, p; \quad k = 1, \ldots, m. \tag{2.11a}$$

onde $$\Delta w_{jk}(t) = \eta \, . \, \delta y_k(t) \, . \, h_j(t) \, . \tag{2.11b}$$

8. Ajuste os pesos v_{ij} entre a camada de entrada e a escondida, segundo as equações seguintes:

$$v_{ij}(t+1) = v_{ij}(t) + \Delta v_{ij}(t), \qquad i = 0, \ldots, n; \qquad j = 1, \ldots, p. \tag{2.12a}$$

onde $$\Delta v_{ij}(t) = \eta \, . \, \delta h_j(t) \, . \, x_i(t) \, . \tag{2.12b}$$

9. Vá ao passo 3 e repita até o passo 8 até que todos os pares do conjunto de treinamento tenham sido apresentados à rede (uma época tenha transcorrido). Repita os passos 3 a 8 por tantas épocas quanto necessário ou desejado.

Como mencionado, o algoritmo backpropagation consiste na *regra delta generalizada* que aplica a técnica de otimização conhecida como *gradiente descendente*. A regra delta generalizada é a expressa pelas equações 2.11b e 2.12b, i.e., a correção de uma sinapse que une o neurônio *i* com o neurônio *j* é dada pelo produto da taxa de aprendizagem η, o gradiente local $\delta\_j(t)$, e o sinal de saída do neurônio *i*. A taxa de aprendizagem η é um fator de escala que nos diz quão rápido devemos nos mover na direção contrária ao gradiente. Valores pequenos conduzem a um aprendizado lento, valores grandes fazem com que a rede oscile e não convirja. Valores usuais para η estão entre 0,3 e 0,8.

Para redes com mais de uma camada escondida, o algoritmo apresentado generaliza-se da seguinte forma: para cada camada adicional insira o cálculo da saída desta camada entre os passos 3 e 4; o cálculo do vetor δ relacionado aos erros da camada adicional entre os passos 6 e 7, e o cálculo de correção das sinapses da camada adicional entre os passos 7 e 8.

A velocidade de aprendizagem da rede pode ser aumentada incluindo-se o fator momento α nas equações de correção das sinapses (Eqs. 2.11 e 2.12). Estas eqs. ficam agora:

$$w_{jk}(t+1) = w_{jk}(t) + \Delta w_{jk}(t), \qquad j = 0,\ldots, p; \qquad k = 1,\ldots, m. \tag{2.13a}$$

$$\Delta w_{jk}(t) = \eta . \delta y_k(t) . h_j(t) + \alpha . \Delta w_{jk}(t-1). \tag{2.13b}$$

$$v_{ij}(t+1) = v_{ij}(t) + \Delta v_{ij}(t), \qquad i = 0,\ldots, n; \qquad j = 1,\ldots, p. \tag{2.14a}$$

$$\Delta v_{ij}(t) = \eta . \delta h_j(t) . x_i(t) + \alpha . \Delta v_{ij}(t-1). \tag{2.14b}$$

O fator momento α atua como uma memória, fazendo com que a direção de correção das sinapses seja uma combinação do gradiente atual e do gradiente anterior. α pode variar entre 0 e 1. Usualmente escolhe-se um valor de α entre 0,5 e 0,9.

Existem outras variações do algoritmo backpropagation, como utilizar taxas de aprendizagem e fatores momento adaptáveis ao longo do treinamento, fazer a correção das sinapses após uma época e não exemplo a exemplo, etc.

2.4.2.2 Quando parar o treinamento da rede

Nas redes backpropagation usadas para classificar e reconhecer padrões, pretende-se alcançar um equilíbrio entre a capacidade da rede de dar respostas corretas aos padrões usados no treinamento e respostas corretas a novos padrões de entrada (balanço entre memorização e generalização). A Fig. 2.15 mostra o efeito de generalização de uma rede neural em função do tempo de treinamento (épocas de aprendizagem). Na figura há duas curvas: a superior mostra o desempenho da rede com o conjunto de treinamento e a inferior o desempenho da rede com o conjunto de teste. Nenhum vetor do conjunto de teste faz parte do conjunto de treinamento (conjuntos disjuntos).

Pode ser visto que após o começo do treinamento e à medida que o mesmo transcorre, o desempenho da rede melhora tanto para o conjunto de treinamento como para o de teste. Para este último, porém, o desempenho é sempre inferior que para o conjunto de treinamento já que contém exemplos que a rede não aprendeu. Após um certo tempo o desempenho praticamente se estabiliza. Continuando o treinamento, o desempenho com o conjunto de treinamento au-

menta mas com o conjunto de teste piora. O que acontece aqui é que a rede está começando a memorizar os exemplos de treinamento e com isto a perder poder de generalização. Se o número de sinapses disponível é suficiente, a rede é capaz de memorizar o conjunto de treinamento inteiro. Isto é contraproducente, não sendo desejável treinar a rede neural até que o erro de treinamento atinja um mínimo.

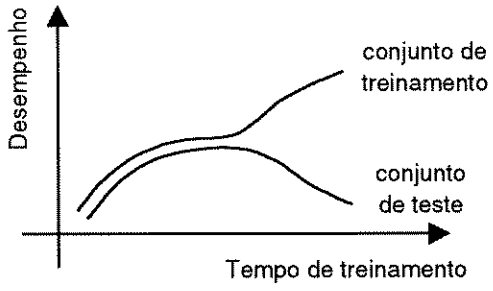


Figura 2.15: Desempenho de uma rede neural em função do tempo de treinamento.

Existem várias técnicas para evitar que a rede perca poder de generalização. Uma delas é suspender o treinamento quando o erro de treino é suficientemente baixo mas não mínimo (0,5 a 1,5%, por exemplo). Outra forma é adicionar pequenas quantidades de ruído aos exemplos de treinamento. O ruído deve ser o suficiente para prevenir a memorização, mas não tão grande que impeça a clusterização dos exemplos de treinamento (confunda a rede). Uma outra forma é diminuir o número de neurônios escondidos de forma a criar um gargalo na rede. Com menos sinapses disponíveis, a rede é obrigada a fazer representações internas compactas de suas entradas. Hecht-Nielsen (1990) sugere, ainda, o uso de dois conjuntos de dados durante o treinamento: um deles chamado de conjunto de treinamento e o outro chamado de treinamento/teste (os conjuntos são disjuntos). O primeiro conjunto é usado para efetivamente corrigir as sinapses da rede. O segundo é usado para testar a rede a intervalos regulares. Quando o desempenho com este segundo conjunto começa a cair, o treinamento é suspenso (tem-se atingido o máximo na curva inferior da Fig. 2.15). Esta forma de determinar quando parar o treinamento da rede é também chamado de validação cruzada.

2.4.3 Redes de Kohonen (*Self-Organizing Maps*)

As redes de Kohonen ou SOM (*Self-Organizing Maps*, Kohonen, 1989), são redes auto-organizáveis que assumem uma estrutura topológica entre os neurônios. Esta é uma propriedade observada no cérebro mas não encontrada em nenhuma outra rede neural artificial.

Nas redes de Kohonen os neurônios são colocados em *arrays* uni ou bidimensionais (mapas de dimensões maiores são possíveis mas pouco usados). Após o treinamento, as posições espaciais dos neurônios no mapa resultante correspondem a caraterísticas intrínsecas dos dados de entrada. A posição espacial de um neurônio é dada pelo seu vetor de pesos (conjunto de sinapses que chega até ele). Adicionalmente, o vetor de pesos de cada neurônio serve como um exemplar dos padrões de entrada associados com o cluster que ele representa.

O treinamento da rede é baseado na competição entre os neurônios. O algoritmo é *não-supervisionado*. Durante o treinamento (processo de aprendizado ou auto-organização da rede), compara-se o padrão de entrada com o vetor de pesos de cada neurônio e aquele cujo vetor de pesos está mais próximo ao de entrada é escolhido como ganhador. Usualmente utiliza-se distância euclidiana para medir esta proximidade. Podem usar-se outras métricas (produto escalar, por exemplo). O neurônio vencedor e seus vizinhos próximos têm, então, seus pesos corrigidos em função do algoritmo de aprendizado. Na medida em que o treinamento avança o número de neurônios vizinhos que têm seus pesos corrigidos vai diminuindo gradualmente. A partir de determinado momento só se corrigem os pesos do neurônio vencedor.

Uma vez treinada, a rede trabalha da seguinte forma: o vetor desconhecido é colocado na entrada e é comparado com os vetores de pesos dos neurônios da rede. Aquele que estiver mais próximo do vetor de entrada é escolhido ganhador e o vetor desconhecido pertence ao cluster que o neurônio ganhador representa. Perceber que nesta rede os neurônios não empregam não-linearidades. De fato não se calculam as saídas dos neurônios como nas redes backpropagation, senão que o funcionamento da rede é baseado na comparação entre o vetor de entrada e os vetores de pesos dos neurônios.

A Fig. 2.16 mostra uma rede de Kohonen com m neurônios, n entradas e matriz de sinapses $\mathbf{W}$ unindo as entradas com os neurônios. Esta rede pode ser utilizada para agrupar um conjunto de vetores contínuos $\mathbf{x} = (x_1,\ldots, x_n)$, em m clusters. Pode ser visto que a topologia de

uma rede de Kohonen é similar à de um perceptron multicamadas com apenas uma camada (Fig. 2.8), distinguindo-se dela por seu funcionamento e algoritmo de treinamento.

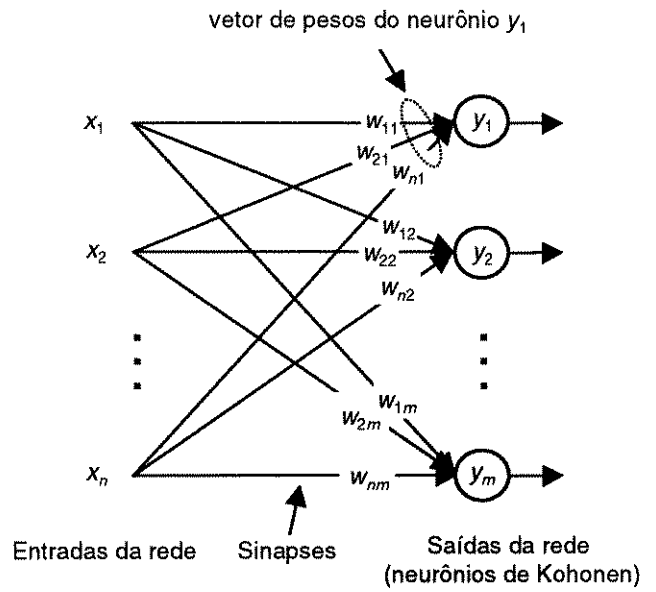


Figura 2.16: Rede de Kohonen com n entradas e m neurônios.

Supondo m =10 neurônios, a Fig. 2.17 mostra um array unidimensional de 10 clusters representando o mapa de Kohonen correspondente. Nesta figura o neurônio vencedor é representado com # e os restantes com * para mostrar os limites das vizinhanças de raio $r = 2$, 1 e 0. As mesmas vizinhanças são mostradas na Fig. 2.18a para mapa retangular e na Fig. 2.18b para mapa hexagonal (cada um com 49 unidades ou neurônios). No mapa retangular cada unidade tem 8 neurônios vizinhos, enquanto que no mapa hexagonal cada neurônio tem 6 neurônios vizinhos. Quando o neurônio ganhador está perto das -ou nas- fronteiras do mapa, o número de unidades vizinhas é menor.

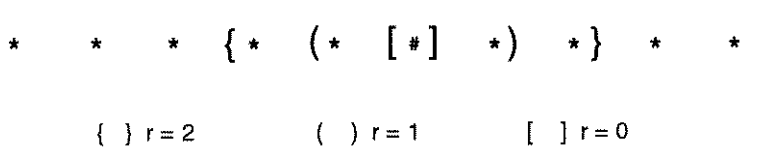


Figura 2.17: Array linear de clusters, mostrando vizinhanças de raio 2, 1 e 0.

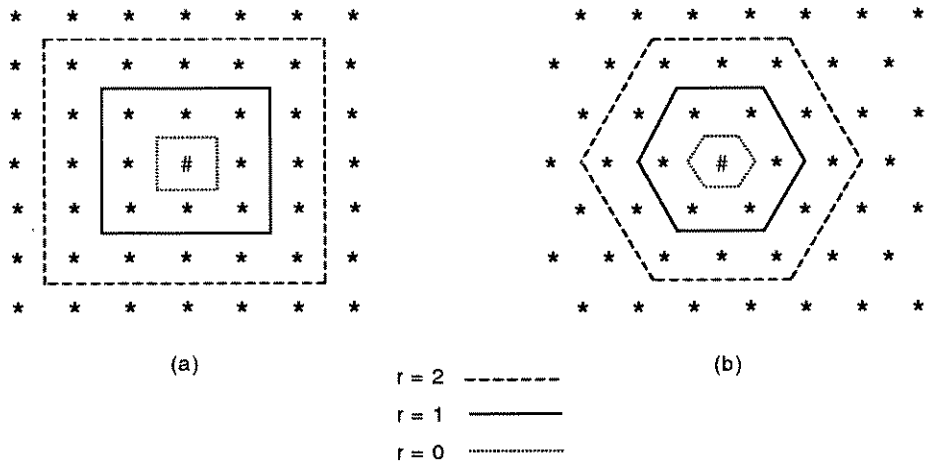


Figura 2.18: Vizinhanças de raio 2, 1 e 0 em (a) array retangular de clusters e (b) array hexagonal de clusters.

2.4.3.1 Algoritmo de treinamento das redes de Kohonen

Nomenclatura a usar

$\mathbf{x}$ vetor de entrada (ou de treinamento); $\mathbf{x} = (x_1, x_2, \ldots, x_n)$.

$\mathbf{w}_j$ vetor de sinapses do neurônio j; $\mathbf{w}_j = (w_{1j}, w_{2j}, \ldots, w_{nj})$.

$D(j)$ distância euclidiana (ou outra função de comparação) entre os vetores $\mathbf{x}$ e $\mathbf{w}_j$.

η taxa de aprendizagem ($0 < \eta(t) < 1$).

r raio de vizinhança do neurônio ganhador.

Λr função *vizinhança*, centrada no neurônio ganhador e de raio r.

Cada neurônio representa um cluster. A localização espacial do cluster é dada pelo vetor de sinapses do neurônio correspondente.

Algoritmo

1. Inicialize os vetores de sinapses $\mathbf{w}_j$ *para* $j = 1, \ldots, m.$, *o raio de vizinhança* r, *e a taxa de aprendizagem* η *(após o algoritmo serão feitas considerações a respeito).*

2. Escolha um vetor do conjunto de treinamento e coloque-o na entrada da rede. Sendo $\mathbf{x}$ *o vetor de entrada, calcule a distância euclidiana entre* $\mathbf{x}$ *e os vetores de sinapses* $\mathbf{w}_j$.

$$D(j) = \sum_{i=1}^{n} (w_{ij} - x_i)^2 \ , \qquad j = 1, \ldots, m.$$

3. Determine o neurônio ganhador encontrando o índice j tal que D(j) é mínimo. Para o neurônio ganhador e para os que pertencem à sua vizinhança (estão dentro de um raio r a partir do neurônio ganhador), corrija as sinapses com a equação seguinte (por ser a equação final, inclui-se nela o fator tempo):

$$w_{ij}(t+1) = w_{ij}(t) + \eta(t)\,[x_i(t) - w_{ij}(t)], \qquad i = 1, \ldots, n. \quad \forall \quad j \in \Lambda r(t) \tag{2.15}$$

Vetorialmente:

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \eta(t)\,[\mathbf{x}(t) - \mathbf{w}_j(t)] \qquad \text{se} \quad j \in \Lambda r(t) \tag{2.16a}$$

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) \qquad \text{se} \quad j \notin \Lambda r(t). \tag{2.16b}$$

4. Repita os passos 2 e 3 até esgotar todos os vetores de treinamento. Diminua então o raio de vizinhança r e a taxa de aprendizagem η.

5. Repita os passos 2 a 4 até que a posição dos neurônios permaneça praticamente inalterada.

Outras opções para a atualização da taxa de aprendizagem η e o raio de vizinhança r podem ser utilizadas. A inicialização dos vetores de sinapses pode ser feita escolhendo-se vetores arbitrários ou vetores aleatórios; a única restrição é não escolher dois ou mais vetores iguais. Quando se tem alguma informação a priori da distribuição dos clusters, as posições iniciais dos vetores podem ser escolhidas aproveitando-se este conhecimento. Em caso de se escolherem posições aleatórias, é desejável que os pesos (componentes de cada vetor) estejam no mesmo intervalo de variação que as componentes dos vetores de entrada (treinamento e teste).

O processo de aprendizado que origina os mapas de Kohonen é estocástico em natureza. Portanto, a exatidão do mapa depende do número de iterações do algoritmo SOM. Adicionalmente, o sucesso na formação do mapa depende de como a taxa de aprendizagem η e a função vizinhança Λr são escolhidas. Não há bases teóricas para esta escolha. Usualmente escolhe-se por tentativa e erro, ou baseando-se em experiências anteriores. Kohonen (1989, 1990a), dá as seguintes sugestões:

1. A taxa de aprendizagem $\eta(t)$, utilizada para adaptar os vetores de sinapses $\mathbf{w}_j(t)$, deve diminuir com o tempo (exemplo a exemplo, época a época, cada p exemplos, etc.). Em particular,

durante as primeiras 1000 iterações $\eta(t)$ deveria começar com um valor próximo de 1 e decrescer monotonicamente. A forma de variação de $\eta(t)$ não é crítica, pode ser linear, exponencial ou inversamente proporcional a t. Por exemplo, usar $\eta(t) = 0{,}9\ (1 - t/1000)$ durante as primeiras 1000 iterações, pode ser uma escolha razoável. É durante esta fase inicial do algoritmo que ocorre o ordenamento topológico dos vetores $\mathbf{w}_j(t)$. Por causa disto esta fase é chamada de *fase de ordenamento*. As iterações restantes do algoritmo são necessárias, principalmente, para o ajuste fino das posições dos vetores. Esta segunda fase é chamada de *fase de convergência*. Para uma boa precisão estatística, durante a fase de convergência $\eta(t)$ deve ser mantida num valor pequeno (da ordem de 0,01 ou menor) por um longo período de tempo (tipicamente, milhares de iterações). Iterações, aqui, não se refere a épocas, mas aos passos 2 e 3 do algoritmo de aprendizado.

2. Para que ocorra o ordenamento topológico dos neurônios, a função vizinhança deve ser cuidadosamente escolhida. Usualmente $\Lambda r(t)$ consiste em uma região quadrada ou hexagonal ao redor do neurônio ganhador (Fig. 2.18a e b). Por exemplo, para um raio de valor 1, a região quadrada inclui o neurônio ganhador e 8 neurônios vizinhos, enquanto que a hexagonal inclui o neurônio ganhador e 6 neurônios vizinhos. Em qualquer caso a função vizinhança $\Lambda r(t)$ deveria começar com um raio que inclua todos os neurônios do mapa e diminuir gradualmente com o tempo (exemplo a exemplo, época a época, cada p exemplos, etc.). Durante as primeiras 1000 iterações do algoritmo, quando ocorre o ordenamento topológico dos vetores de sinapses, o raio r pode ser diminuído linearmente até atingir o valor 1. Durante a fase de convergência do algoritmo, $\Lambda r(t)$ deve conter unicamente o neurônio ganhador ($r = 0$) ou, como máximo, o neurônio ganhador e seus vizinhos imediatos ($r = 1$).

2.4.4 Redes de Kohonen-Grossberg (*Counterpropagation Networks*)

As redes de Kohonen-Grossberg ou *counterpropagation networks*, desenvolvidas por Hecht-Nielsen (1987b, 1987c, 1988), são redes multicamadas baseadas na combinação de dois algoritmos: os mapas de Kohonen e os *outstar* de Grossberg (1969). As redes constam de uma

camada de entrada, uma de clusterização e uma de saída, e podem ser usadas para comprimir dados, aproximar funções, associar/classificar padrões, etc.

A Fig. 2.19 mostra a arquitetura de uma rede de Kohonen-Grossberg do tipo *feedforward only*. Pode-se ver que sua topologia é similar à de uma rede backpropagation, distinguindo-se dela por seu funcionamento e algoritmo de treinamento. Como nas redes de Kohonen, as redes de Kohonen-Grossberg não utilizam não-linearidades em seus neurônios.

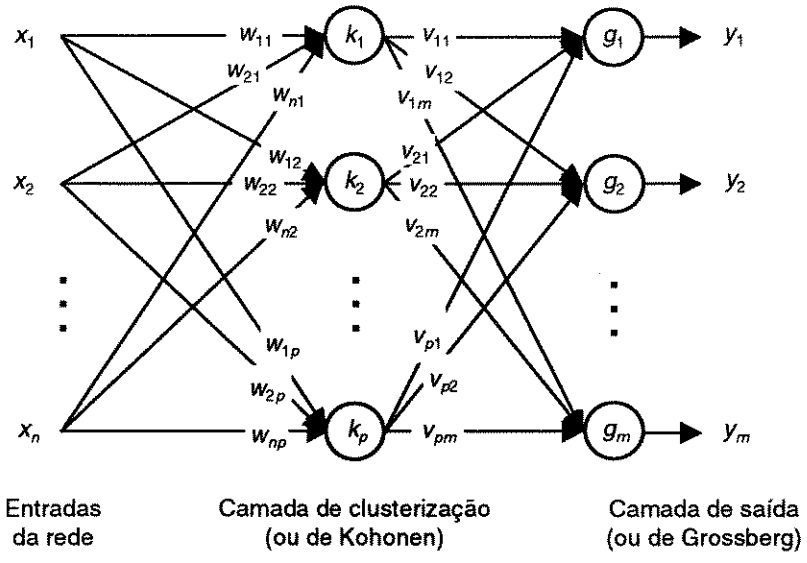


Figura 2.19: Rede "Feedforward Only Counterpropagation"

2.4.4.1 Algoritmo de treinamento das redes de Kohonen-Grossberg

O algoritmo de treinamento das redes de Kohonen-Grossberg consta de duas fases. Na primeira fase é realizada a clusterização dos vetores de entrada como nos mapas de Kohonen. Nesta fase é como se a camada de saída não existisse, adaptando-se unicamente os pesos da matriz **W**. O algoritmo utilizado é o SOM (*self-organizing maps*), portanto o treinamento é *não-supervisionado*. As eqs. utilizadas são as mesmas da Seção 2.4.3.1.

Na segunda fase é treinada a camada de saída da rede (matriz **V**). Esta camada tem a função de mapear a saída da camada de Kohonen no vetor de saída desejado. Assim, para esta fase utilizam-se pares de vetores de treinamento (*entrada/saída-desejada*) e o treino passa a ser *supervisionado*. Nesta fase coloca-se um vetor na entrada da rede e determina-se o neurônio ga-

nhador da camada de Kohonen. A saída deste neurônio é fixada em 1 e a dos outros neurônios da camada em 0 (este tipo de funcionamento denomina-se *o ganhador leva tudo* ou *winner takes all*). As sinapses de **V** são então corrigidas em função do vetor de saída desejado **d**. Como somente um neurônio da camada de Kohonen tem saída 1, só as sinapses que unem este neurônio com a camada de saída são corrigidas. Supondo que o neurônio ganhador da camada de Kohonen seja o i, a equação utilizada para corrigir **V** é a seguinte:

$$v_{ij}(t+1) = v_{ij}(t) + \beta(t)\,[d_j(t) - v_{ij}(t)], \quad j = 1,\ldots, m. \quad i\text{: neurônio ganhador} \qquad (2.17)$$

Apresenta-se, então, à rede um novo par de vetores de treinamento e a correção de **V** se repete. Após apresentar à rede todos os pares de treinamento, uma época terá transcorrido. O processo continua por tantas épocas quanto necessário.

β é a taxa de aprendizagem da camada de saída. Usualmente inicia-se o treinamento com β igual a 0,1, reduzindo este valor gradualmente à medida que o treinamento é realizado (após cada exemplo, após cada época, a cada p exemplos, etc.).

Em operação normal, a saída da rede é computada multiplicando-se o vetor de saída da camada de Kohonen pela matriz de sinapses **V**. Matematicamente $\mathbf{y} = \mathbf{k}.\mathbf{V}$, onde **y** é o vetor de saída da rede e **k** o vetor de saída da camada de Kohonen.

2.4.5 Redes LVQ (*Learning Vector Quantization*)

Learning vector quantization (Kohonen, 1989, 1990b; Fausett, 1993), é um método de classificação de padrões, no qual cada unidade ou neurônio de saída representa uma classe ou categoria em particular (vários neurônios de saída podem ser usados para representar uma mesma classe). O vetor de pesos de um neurônio de saída é chamado de *vetor de referência* ou *codebook vector* da classe que representa. Durante o treinamento, os neurônios são posicionados ajustando seus pesos através de um aprendizado supervisionado, de forma a aproximar as superfícies de decisão das superfícies teóricas do classificador de Bayes. Para o treinamento é necessário que um conjunto de padrões com classificações conhecidas esteja disponível, além

de uma distribuição inicial dos vetores de referência cada um representando uma classificação conhecida.

Após o treinamento a rede LVQ classifica um vetor de entrada atribuindo-o à classe do neurônio de saída que tem seu vetor de pesos, ou vetor de referência, mais próximo dele.

A Fig. 2.20 mostra a arquitetura de uma rede neural LVQ. Pode ser visto que, essencialmente, é a mesma arquitetura que a dos mapas de Kohonen (porém, não se assume uma estrutura topológica entre os neurônios de saída). Cada neurônio de saída representa uma classe ou categoria conhecida.

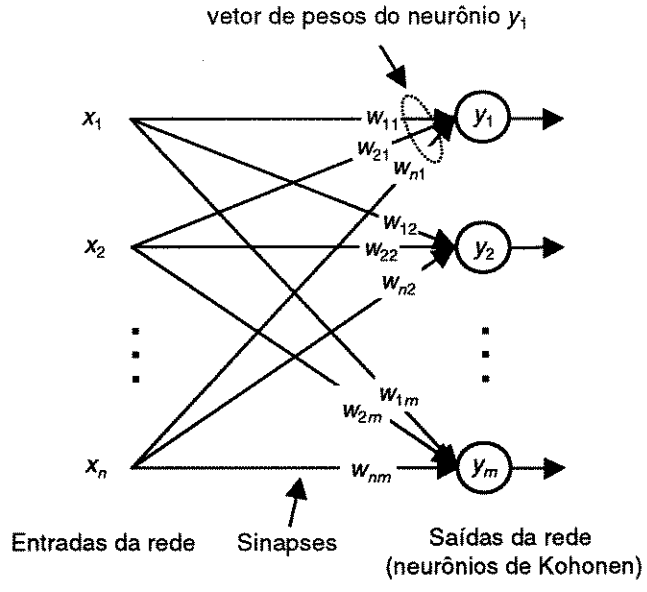


Figura 2.20: Rede neural LVQ (Learning Vector Quantization).

2.4.5.1 Algoritmo de treinamento LVQ1

Nomenclatura a usar

$\mathbf{x}$	vetor de treinamento; $\mathbf{x} = (x_1, x_2, \ldots, x_n)$.
T	classe ou categoria correta do vetor de treinamento.
$\mathbf{w}_j$	vetor de pesos do j-ésimo neurônio de saída; $\mathbf{w} = (w_{1j}, w_{2j}, \ldots, w_{nj})$.
C_j	classe ou categoria representada pelo j-ésimo neurônio de saída.
$\|\mathbf{x} - \mathbf{w}_j\|$	distância euclidiana entre o vetor de entrada e o vetor de pesos do j-ésimo neurônio de saída.

η taxa de aprendizagem ($0 < \eta(t) < 1$).

Como mencionado, o algoritmo LVQ tem por objetivo determinar o neurônio de saída (vetor de referência) que está mais perto do vetor de entrada. Durante o aprendizado se $\mathbf{x}$ e $\mathbf{w}_c$ (vetor de referência mais perto de $\mathbf{x}$) pertencem à mesma classe, move-se $\mathbf{w}_c$ em direção de $\mathbf{x}$; se $\mathbf{x}$ e $\mathbf{w}_c$ pertencem a classes diferentes, move-se o vetor de pesos $\mathbf{w}_c$ de forma a separá-lo do vetor $\mathbf{x}$.

Algoritmo

1. Inicialize os vetores de referência $\mathbf{w}_j$ *para* $j = 1,\ldots, m$, *e a taxa de aprendizagem* η *(após o algoritmo são discutidas várias formas de fazer isto).*

2. Para cada vetor de treinamento $\mathbf{x}$ *encontre o neurônio* j *tal que* $\|\mathbf{x} - \mathbf{w}_j\|$ *é mínimo e, então, atualize o vetor* $\mathbf{w}_j$ *como segue:*

$$Se\ T = C_j\ , \qquad \mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \eta(t).[\mathbf{x}(t) - \mathbf{w}_j(t)]; \tag{2.18a}$$

$$Se\ T \neq C_j\ , \qquad \mathbf{w}_j(t+1) = \mathbf{w}_j(t) - \eta(t).[\mathbf{x}(t) - \mathbf{w}_j(t)]. \tag{2.18b}$$

Os neurônios restantes ficam inalterados.

3. Reduza a taxa de aprendizagem η.

4. Repita os passos 2 e 3 por tantas épocas quanto necessário. A condição para acabar o treinamento é ter alcançado um número de épocas determinado ou a taxa de aprendizagem ter atingido um valor suficientemente pequeno.

A forma mais simples de inicializar os vetores de referência, é tomar os primeiros m vetores de treinamento (cada um representando uma classificação conhecida e diferente), e usá-los como vetores $\mathbf{w}_j$. Outra forma possível é colocar os vetores em posições aleatórias e utilizar o algoritmo SOM (*self-organizing maps*) para definir suas posições iniciais. Após a fase SOM aplica-se o algoritmo LVQ.

Quanto à taxa de aprendizagem η, ela deve inicializar-se com um valor pequeno (0,01 ou 0,02), e decrescer até zero monotonicamente com o tempo (por exemplo, em cem mil itera-

ções). A lei de decrescimento pode ser linear e não é necessário esperar que chegue até 0 para suspender o treinamento.

Os algoritmos seguintes, LVQ2, LVQ2.1 e LVQ3 (Kohonen, 1990a e 1990b; Fausett, 1993), são aperfeiçoamentos do algoritmo original LVQ1. No algoritmo original somente o vetor de referência que está mais perto do vetor de entrada é atualizado. A direção em que é movido depende dele pertencer ou não à mesma classe do vetor de entrada. Nos algoritmos seguintes dois vetores –o ganhador e o segundo mais perto do vetor de entrada– são atualizados, sempre que algumas condições forem satisfeitas. A idéia é que se o vetor de entrada está aproximadamente à mesma distância de dois vetores de referência, então que ambos vetores de referência possam aprender.

2.4.5.2 Algoritmo de treinamento LVQ2

Seja:

$\mathbf{x}$ o vetor de entrada;

$\mathbf{y}_c$ o vetor de referência que está mais perto de $\mathbf{x}$;

$\mathbf{y}_r$ o vetor de referência que, depois de $\mathbf{y}_c$, está mais perto de $\mathbf{x}$;

d_c a distância de $\mathbf{x}$ a $\mathbf{y}_c$;

d_r a distância de $\mathbf{x}$ a $\mathbf{y}_r$;

Sejam as distâncias d_c e d_r aproximadamente iguais. Esta condição é expressa definindo-se uma janela em função de um parâmetro $\varepsilon > 0$, tal que $\mathbf{x}$ pertence a esta janela se:

$$\frac{d_c}{d_r} > 1 - \varepsilon \qquad \text{e} \qquad \frac{d_r}{d_c} < 1 + \varepsilon. \tag{2.19}$$

O valor de ε depende do número de exemplos de treinamento; se há poucos exemplos usa-se um ε grande; se há muitos exemplos usa-se um ε pequeno. Um valor típico utilizado é ε=0,35.

No algoritmo LVQ2, os vetores $\mathbf{y}_c$ e $\mathbf{y}_r$ são corrigidos se o vetor de entrada $\mathbf{x}$ pertence à janela definida, $\mathbf{y}_c$ e $\mathbf{y}_r$ pertencem a classes diferentes e $\mathbf{x}$ pertence à mesma classe de $\mathbf{y}_r$. Se estas condições se verificam, então $\mathbf{y}_c$ e $\mathbf{y}_r$ são corrigidos com as seguintes equações:

$$\mathbf{y}_c(t+1) = \mathbf{y}_c(t) - \eta(t).[\mathbf{x}(t) - \mathbf{y}_c(t)]; \tag{2.20a}$$

$$\mathbf{y}_r(t+1) = \mathbf{y}_r(t) + \eta(t).[\mathbf{x}(t) - \mathbf{y}_r(t)]. \tag{2.20b}$$

Desta forma, $\mathbf{y}_r$ é aproximado do vetor **x** enquanto que $\mathbf{y}_c$ é distanciado dele.

2.4.5.3 Algoritmo de treinamento LVQ2.1

Nesta modificação, Kohonen considera os dois vetores de referência mais próximos de **x** como $\mathbf{y}_{c1}$ e $\mathbf{y}_{c2}$. A condição para atualizar estes vetores é que um deles, por exemplo $\mathbf{y}_{c1}$, pertença à mesma classe de **x** e que o outro, $\mathbf{y}_{c2}$, não pertença. O algoritmo LVQ2.1 não distingue se o vetor mais perto de **x** representa a classe correta ou a incorreta. Assim como no algoritmo LVQ2, é necessário que **x** pertença à janela definida por ε para que a correção dos vetores ocorra. O teste para saber se **x** pertence à janela é dado agora por:

$$\min\left[\frac{d_{c1}}{d_{c2}}, \frac{d_{c2}}{d_{c1}}\right] > 1-\varepsilon \qquad \text{e} \qquad \max\left[\frac{d_{c1}}{d_{c2}}, \frac{d_{c2}}{d_{c1}}\right] > 1+\varepsilon. \tag{2.21}$$

As expressões são mais complexas que no algoritmo LVQ2, devido ao fato de não se saber se **x** está mais perto de $\mathbf{y}_{c1}$ ou de $\mathbf{y}_{c2}$. Se as condições são satisfeitas, o vetor de referência que pertence à mesma classe que **x** é corrigido com a seguinte equação:

$$\mathbf{y}_{c1}(t+1) = \mathbf{y}_{c1}(t) + \eta(t).[\mathbf{x}(t) - \mathbf{y}_{c1}(t)]; \tag{2.22a}$$

O vetor de referência que não pertence à mesma classe de **x** é corrigido com a equação seguinte:

$$\mathbf{y}_{c2}(t+1) = \mathbf{y}_{c2}(t) - \eta(t).[\mathbf{x}(t) - \mathbf{y}_{c2}(t)]. \tag{2.22b}$$

Assim, $\mathbf{y}_{c1}$ é aproximado de **x** enquanto $\mathbf{y}_{c2}$ é distanciado dele.

2.4.5.4 Algoritmo de treinamento LVQ3

Um refinamento posterior, o algoritmo LVQ3, permite que os dois vetores de referência mais próximos de **x** sejam corrigidos ainda que ambos pertençam à mesma classe de **x**. Como antes **x** deve pertencer a uma janela, neste caso definida por:

$$\min\left[\frac{d_{c1}}{d_{c2}}\ ,\ \frac{d_{c2}}{d_{c1}}\right] > (1-\varepsilon)\,(1+\varepsilon). \tag{2.23}$$

onde um valor de $\varepsilon = 0{,}2$ é indicado. Se um dos dois vetores mais próximos de $\mathbf{x}$, $\mathbf{y}_{c1}$ –por exemplo–, pertence à mesma classe que $\mathbf{x}$ e o outro vetor, $\mathbf{y}_{c2}$, pertence a uma classe diferente, os vetores são corrigidos utilizando as eqs. do algoritmo LVQ2.1. Se, no entanto, ambos vetores pertencem à mesma classe que $\mathbf{x}$, os vetores são corrigidos utilizando a seguinte equação:

$$\mathbf{y}_c(t+1) = \mathbf{y}_c(t) + \beta(t).[\mathbf{x}(t) - \mathbf{y}_c(t)]. \tag{2.24}$$

Esta equação vale tanto para $\mathbf{y}_{c1}$ como para $\mathbf{y}_{c2}$. A taxa de aprendizagem $\beta(t)$ é um submúltiplo da taxa de aprendizagem $\eta(t)$ utilizada quando $\mathbf{y}_{c1}$ e $\mathbf{y}_{c2}$. pertencem a classes diferentes. Tipicamente utiliza-se um fator de 0,1 a 0,5 para submúltiplo, com os valores mais pequenos correspondendo a janelas mais estreitas. Matematicamente:

$$\beta(t) = q.\eta(t), \qquad \text{para} \quad 0{,}1 < q < 0{,}5. \tag{2.25}$$

Esta modificação no processo de aprendizagem permite que os vetores de referência representem cada vez melhor a distribuição de classes, evitando que se distanciem de suas posições ótimas quando o aprendizado é muito longo.

Tanto as redes de Kohonen, como as de Kohonen-Grossberg e as LVQ, são, no fundo, quantizadores vetoriais, onde o número de vetores do *codebook* é arbitrário, e não necessariamente uma potência de dois como nos quantizadores vetoriais tradicionais. Nas redes de Kohonen e Kohonen-Grossberg, o número de vetores do *codebook* é dado pelo número de neurônios da camada de Kohonen; nas redes LVQ, pelo número de neurônios da rede.

Capítulo 3

Parâmetros de Entrada

3.1 Base de dados

Os sistemas de reconhecimento de fala de que trata este trabalho são independentes do locutor. Nos sistemas de reconhecimento de fala independentes do locutor, é necessário dispor de uma base de dados suficientemente grande e variada para treinar o sistema. O objetivo de utilizar uma base de dados assim, é incluir na mesma diferentes formas de pronunciar as palavras a reconhecer (raramente uma mesma palavra é pronunciada duas vezes da mesma forma, mesmo palavras simples e pelo mesmo falante). Assim, é necessário que a base de dados inclua pessoas com diferentes sotaques, faixas etárias, culturas, sexos, idades, temperamentos, etc. Quanto maior for a variedade de pessoas incluídas na base de dados de treinamento do sistema de reconhecimento, melhor será, em geral, o desempenho do mesmo.

Também é desejável que o ambiente e as condições de aquisição da base de dados sejam similares àquelas onde o sistema de reconhecimento será utilizado. Isto permite incluir no material de treinamento os ruídos de fundo que existirão na prática (conversas alheias, ruídos de máquinas, tráfego urbano, etc.).

Na época de início desta tese, não existia na Unicamp – nem, pelo que nos consta, no Brasil – uma base de dados em língua portuguesa que pudesse ser utilizada para treinamento e teste dos sistemas. A primeira providência tomada foi a criação de uma base de dados mínima com a qual fosse possível levar adiante a pesquisa. Para isto definiram-se 4 vocabulários a utilizar ao longo do trabalho. Eles foram os seguintes:

1. ***Dígitos***: contendo as palavras *zero, um, dois, três, quatro, cinco, seis, sete, oito* e *nove*.

2. ***Comandos de Cálculo***: contendo 11 palavras que permitem, por exemplo, comandar uma calculadora vocalmente. Este vocabulário é formado pelas palavras: *ponto, vírgula, negativo, cancele, vezes, divida-por, mais, menos, igual, inicie* e *calcule.*

3. ***Comandos de Movimento***: contendo 11 palavras que permitem comandar o movimento de um dispositivo, por exemplo, um robô. Este vocabulário é formado pelas palavras: *esquerda, direita, cima, baixo, frente, trás, pare, siga, rápido, devagar* e *mova-se.*

4. ***A união dos três anteriores***: formando um vocabulário de 32 palavras.

A etapa de criação da base de dados demorou alguns meses. Reuniram-se 30 locutores provenientes de diferentes regiões do pais, 15 homens e 15 mulheres. Após triagem das gravações realizadas aproveitaram-se 24 locutores, 12 homens e 12 mulheres. De cada locutor gravaram-se 4 elocuções das palavras do vocabulário um (dígitos), e 3 elocuções das palavras dos vocabulários dois e três (comandos de cálculo e de movimento). Este número de locutores é muito pequeno se comparado ao que seria desejável (mínimo de 100 locutores). Assim, para melhor aproveitamento da base de dados recorreu-se ao seguinte artifício: dividiu-se a base em três conjuntos disjuntos com 8 locutores em cada um deles (4 homens e 4 mulheres). Denominando estes conjuntos de A, B e C, tem-se que para cada sistema sob pesquisa, realizaram-se 3 simulações:

Simulação 1: conjunto de treinamento = A + B; conjunto de teste = C.

Simulação 2: conjunto de treinamento = A + C; conjunto de teste = B.

Simulação 3: conjunto de treinamento = B + C; conjunto de teste = A.

O desempenho final do sistema foi medido calculando-se a média das três simulações anteriores (em quase todos os casos, o resultado das três simulações foi o mesmo). Pode ver-se que em cada simulação foram utilizados 16 locutores para o treinamento do sistema (8 homens e 8 mulheres), e 8 locutores para o teste (4 homens e 4 mulheres). Nenhum dos locutores utilizado nos testes do sistema participou do treinamento do mesmo garantindo, assim, que os resultados obtidos corresponderam aos de um sistema independente do locutor.

Os sinais de voz originais correspondentes às palavras gravadas foram limitados em faixa a 3,4 kHz, amostrados a 8 kHz e quantizados com 12 bits/amostra. Esta limitação na faixa de freqüências foi devido à intenção de utilizar o sistema de reconhecimento de fala na rede telefônica. A

gravação das palavras foi feita em ambiente normal utilizando-se microfone dinâmico, resultando em sinais com relação sinal/ruído (SNR) superior a 50 dB. Posteriormente foi feita a detecção manual do início e fim das palavras, para se obter uma avaliação do sistema de reconhecimento dependente, apenas, do desempenho da rede neural. Com isto, possíveis erros em um sistema automático de detecção de início e fim das palavras não mascaram o resultado das simulações da rede.

Após a detecção de início e fim de cada palavra, o sinal de fala foi pré-enfatizado com um coeficiente de pré-ênfase de 0,9 e analisado utilizando janelas de Hamming ou de Hanning. A pré-ênfase é realizada para compensar a queda de 6 dB/oitava causada pelo efeito combinado dos pulsos glotais (–12 dB/oitava) e de irradiação nos lábios (+6 dB/oitava). Simulações prévias realizadas com sinais sem pré-ênfase, resultaram numa queda sensível do desempenho do sistema.

A expressão matemática das janelas utilizadas é a seguinte:

Janela de Hamming:

$$w(n) = \begin{cases} 0{,}54 - 0{,}46 \cos\left(\dfrac{2\pi n}{N-1}\right) & , \quad 0 \le n \le N-1 \\ 0 & , \quad \text{para os outros } n \end{cases} \tag{3.1}$$

Pode ser visto que a amplitude nos extremos da janela cai a 8% de sua amplitude máxima. A amplitude máxima ocorre em $n = (N-1)/2$ (centro da janela).

Janela de Hanning:

$$w(n) = \begin{cases} \dfrac{1}{2}\left[1 - \cos\left(\dfrac{2\pi n}{N-1}\right)\right] & , \quad 0 \le n \le N-1 \\ 0 & , \quad \text{para os outros } n \end{cases} \tag{3.2}$$

Nesta janela, a amplitude nos extremos cai a 0. Esta característica é desejável em alguns tipos de análise, como será visto no Cap. 4, Seção 4.5.2.

Para o teste dos sistemas com sinais ruidosos, adicionou-se ruído branco gaussiano aos sinais de fala de forma a atingir relações sinal/ruído de 20 dB e 10 dB. Assim, no total, trabalhou-se com 2544 palavras sem ruído, 2544 palavras com SNR= 20 dB e 2544 palavras com SNR= 10 dB.

A geração do ruído branco gaussiano foi feita da seguinte forma: sendo x uma variável aleatória uniformemente distribuída entre $-\Delta_x/2$ e $\Delta_x/2$, tem-se que a média e a variância de x são dadas por:

$$\text{média de } x = m_x = 0. \tag{3.3}$$

$$\text{variância de } x = \sigma_x^{\,2} = \Delta_x^{\,2} / 12. \tag{3.4}$$

Considerando-se o *Teorema do Limite Central*, pode-se dizer que a soma de 12 variáveis aleatórias uniformemente distribuídas entre $-\Delta_x/2$ e $\Delta_x/2$ resultará em um sinal $r(n)$ com distribuição praticamente gaussiana, de média 0 e variância Δ_x^2. Isto é:

$$\text{média de } r = m_r = 0. \tag{3.5}$$

$$\text{variância de } r = \sigma_r^{\,2} = \Delta_x^{\,2}. \tag{3.6}$$

Por outro lado, a potência de ruído necessária, $\sigma_r^{\,2}$, para que adicionado a um sinal $s(n)$ de potência $\sigma_s^{\,2}$ resulte em um sinal com relação sinal/ruído SNR, é dada por:

$$\sigma_r^{\,2} = \sigma_s^{\,2} \,.\, 10^{-\mathrm{SNR}/10}. \tag{3.7}$$

onde

$$\sigma_s^2 = \frac{1}{M} \sum_{n=0}^{M-1} s^2(n). \tag{3.8}$$

M corresponde ao número de amostras do sinal. Relacionando as Eqs. 3.6 e 3.7, é possível calcular o valor Δ_x necessário à variável x, para poder gerar o ruído $r(n)$ com a distribuição gaussiana desejada. Isto é:

$$\Delta_x = [\sigma_r^{\,2}]^{1/2} = [\sigma_s^{\,2} \,.\, 10^{-\mathrm{SNR}/10}]^{1/2} \tag{3.9}$$

Somando amostra a amostra o sinal $s(n)$ e o ruído gerado $r(n)$, obtém-se o sinal com a relação sinal/ruído desejada.

3.2 Análise do sinal de fala

Quanto ao tipo de análise realizada nos sinais de fala, deve mencionar-se primeiramente que o principal problema que possuem as redes neurais para trabalhar em reconhecimento de fala, é sua limitação para lidar eficientemente com a estrutura temporal destes sinais. Uma mesma palavra pode ser pronunciada mais rápida ou mais lentamente e a rede neural tem que absorver estas variações.

Em vocabulários que misturam palavras longas e curtas este problema se agrava, já que além das diferentes durações nas elocuções de uma mesma palavra, existe agora o problema de haver palavras de duração inerentemente diferente.

Se o número de entradas da rede neural é diretamente proporcional ao número de quadros de análise utilizado, e se uma mesma rede neural é utilizada para reconhecer todas as palavras do vocabulário em questão, então deve implementar-se algum mecanismo que mantenha constante o número de entradas da rede para todas as palavras. Para isto faz-se necessário:

1. *Analisar todas as palavras com igual número de quadros.* Isto pode causar que os quadros das palavras mais longas superem os 40 ms o que tornaria os trechos analisados não estacionários (a análise espectral assume que o trecho sob análise é ergódico e estacionário). Ao analisar todas as palavras com igual número de quadros, quadros de diferentes elocuções de uma mesma palavra tendem a corresponder aos mesmos sons.

2. *Analisar as palavras com quadros de comprimento fixo (de até 40 ms)*, e depois:

   a) colocar o número de entradas da rede suficientemente grande de forma que caiba a palavra mais longa e para as palavras menores completar as entradas vazias da rede com zeros;

   b) dizimar ou interpolar o número de quadros de análise de cada palavra de forma que todas fiquem com igual número de quadros (uma contribuição importante desta tese está no método utilizado para realizar esta dizimação/interpolação).

Visto que neste trabalho implementaram-se todas as opções mencionadas acima, temos que:

- Para o caso de se analisarem todas as palavras com igual número de quadros, foi testado o uso de 40 a 80 quadros de análise por palavra, com superposição de 1/3 entre quadros consecutivos. Assim, palavras mais longas tiveram quadros mais longos. Com a base de dados utilizada e com 40 quadros de análise por palavra, a duração dos quadros variou entre 10,8 e 42 ms.

- Para o caso de se analisarem as palavras com quadros de comprimento fixo, utilizaram-se quadros de 30 ms recalculados a cada 20 ms. Nas técnicas *Trace Segmentation* e *Individual Trace Segmentation*, utilizaram-se quadros de análise de 30 ms recalculados a cada 4 ms e a cada 2 ms.

Após realizar-se a pré-ênfase e o janelamento do sinal de fala, foram calculados os parâmetros espectrais e temporais utilizados como entradas da rede neural.

Considerando que cada quadro de análise gera um vetor de parâmetros espectrais e/ou temporais, o sinal de fala fica representado por uma seqüência temporal de vetores. O número original de vetores obtidos é igual ao número de quadros de análise utilizado. Para a análise com quadros de duração fixa e fazendo-se a análise a cada 20 ms, obtêm-se 50 vetores por segundo. O número de componentes de cada vetor é função do número de parâmetros calculados. O número de entradas da rede neural será o produto do número de vetores empregado pelo número de componentes de cada vetor.

A próxima seção discute os diferentes parâmetros utilizados neste trabalho e os algoritmos implementados para a obtenção dos mesmos.

3.3 Parâmetros utilizados como entradas da rede neural

Como foi mencionado, a análise do sinal de fala é feita computando-se parâmetros espectrais e/ou temporais a partir do sinal acústico da fala, parâmetros estes que devem permitir a discriminação e classificação das diferentes palavras ou sons a reconhecer.

A análise espectral pode-se realizar utilizando bancos de filtros, análise LPC (*Linear Prediction Coding*), ou DFT (*Discrete Fourier Transform*). Os parâmetros espectrais usualmente calculados são: coeficientes LPC, de reflexão, cepstrais, mel-cepstrais, etc. Na análise temporal, os parâmetros usualmente calculados são: taxa de cruzamentos por zero, perfil de energia do sinal, pitch, aproximações das derivadas temporais dos parâmetros espectrais e temporais anteriores, como –por exemplo– parâmetros delta-cepstrais, delta-mel-cepstrais, delta-energia, etc.

Os parâmetros resultantes da análise do sinal podem se empregar individualmente, para alimentar o sistema de reconhecimento, ou podem se combinar entre si. O usual é empregar mais de um parâmetro como entrada do sistema, visando melhorar as taxas de acerto do mesmo (Runstein et al. 1995).

Neste trabalho foram calculados e testados os seguintes parâmetros:

- ***ESPECTRAIS:*** coeficientes LPC, de reflexão, cepstrais, mel-cepstrais e mel-cepstrais com subtração da média espectral.

- ***TEMPORAIS:*** perfil de energia da palavra, sonoridade, delta-energia, delta-cepstrais e delta-mel-cepstrais.

Para o cálculo dos diferentes parâmetros utilizaram-se os seguintes algoritmos:

- Coeficientes LPC e de Reflexão

Foram obtidos pelo método da autocorrelação, utilizando o algoritmo recursivo de Levinson-Durbin (Rabiner & Schafer, 1978). O número mínimo de coeficientes LPC a calcular deve ser maior ou igual a fs/1000, onde fs é a freqüência de amostragem. Considerando que fs = 8 kHz, são necessários, como mínimo, o cômputo de 8 coeficientes LPC. Esta exigência é necessária para que o modelo de tubos concatenados que dá origem à teoria LPC seja válido (Rabiner & Schafer, 1978). Em nosso caso foram calculados 12 coeficientes LPC por quadro, para que os 4 pólos extras modelassem os zeros responsáveis pelos sons nasalizados.

- Coeficientes cepstrais

Foram testados dois tipos de coeficientes cepstrais: os obtidos a partir dos coeficientes LPC (Rabiner & Schafer, 1978), e os calculados via DFT com o seguinte algoritmo (Rabiner & Schafer, 1978; Davis & Mermelstein, 1980; Picone, 1993):

- DFT das amostras pertencentes ao quadro de análise;
- cálculo do log do módulo da DFT;
- cálculo da transformada DFT inversa, obtendo-se os coeficientes cepstrais.

Resumidamente: $\text{ceps} = \text{DFT}^{-1}\ (\ \log\ |\ \text{DFT(amostras)}\ |\ )$.

Na prática utilizou-se uma FFT (*Fast Fourier Transform*) de 512 pontos para o cálculo da DFT. Mesmo assim, o cálculo dos coeficientes cepstrais via FFT demora, aproximadamente, 3 vezes mais que quando são calculados via LPC. Foram calculados 13 coeficientes cepstrais por quadro. O primeiro não é utilizado devido a representar o valor médio da potência do sinal, sendo este um parâmetro inútil em reconhecimento de fala.

Os coeficientes cepstrais obtidos via FFT tem a seguinte característica: os de ordem baixa carregam informação do perfil do espectro do sinal, enquanto que os de ordem alta carregam

informação da excitação. Em reconhecimento de fala normalmente se utilizam só os de ordem baixa (menor que 20).

- Coeficientes mel-cepstrais

Os coeficientes mel-cepstrais são equivalentes aos coeficientes cepstrais filtrados por um banco de filtros na escala *mel*. A escala mel tenta levar em conta a resposta do ouvido humano aos diferentes sons, mapeando a freqüência percebida (*pitch*) de um tom, em uma escala linear. Estritamente falando, pitch é o correlato perceptual da freqüência de vibração das cordas vocais (F_0). O'Shaughnessy (1987) dá a seguinte expressão para mapear a freqüência real f em Hz, na freqüência percebida (*pitch*) em mels:

$$\textit{freq. percebida (pitch) em mels} = 2595 \,.\, \log_{10}(1 + f\,(\text{Hz}) / 700). \tag{3.10}$$

A tabela 3.1 mostra as freqüências centrais e larguras de banda (BW), dos primeiros 20 filtros na escala mel (Picone, 1993). Neste trabalho utilizaram-se só os primeiros 18 filtros da tabela, uma vez que os sinais de fala foram limitados em faixa a 3400 Hz.

Tabela 3.1: Freqüências centrais e larguras de banda dos primeiros 20 filtros na escala mel.

Banda	Freq. Central (Hz)	BW (Hz)	Banda	Freq. Central (Hz)	BW (Hz)
1	100	100	11	1149	160
2	200	100	12	1320	184
3	300	100	13	1516	211
4	400	100	14	1741	242
5	500	100	15	2000	278
6	600	100	16	2297	320
7	700	100	17	2639	367
8	800	100	18	3031	422
9	900	100	19	3482	484
10	1000	124	20	4000	556

Quanto aos coeficientes mel-cepstrais, foram testados os obtidos por três algoritmos diferentes: o descrito por Davis & Mermelstein (1980), o descrito por Deller, Proakis e Hansen (1993), e o obtido através da transformação bilinear (Picone, 1993).

Algoritmo de Davis & Mermelstein:

- FFT das amostras pertencentes ao quadro de análise;

- cálculo do quadrado do módulo da FFT;
- filtragem do sinal acima por um banco de filtros triangulares na escala Mel;
- cálculo do log da energia na saída dos filtros;
- cálculo da DCT (Discrete Cosine Transform), obtendo-se os coeficientes Mel.

Algoritmo de Deller, Proakis & Hansen:

- FFT das amostras pertencentes ao quadro de análise;
- cálculo do log do quadrado do módulo da FFT;
- filtragem do sinal acima por um banco de filtros triangulares na escala Mel;
- cálculo da DFT inversa obtendo-se os coeficientes Mel.

As diferenças deste algoritmo com o de Davis & Mermelstein são o cálculo do log da energia antes de passar pelos filtros na escala Mel, e o uso de DFT^{-1} ao invés de DCT.

Transformação bilinear:

Este método obtém os coeficientes mel-cepstrais fazendo um cálculo recursivo sobre os coeficientes cepstrais (Picone, 1993). O cálculo baseia-se na transformação bilinear definida por:

$$2\pi f_t = 2\pi \frac{f}{f_s} + 2 \tan^{-1}\left(\frac{\alpha_{bt} sen\left(2\pi \frac{f}{f_s}\right)}{1-\alpha_{bt}\cos\left(2\pi \frac{f}{f_s}\right)} \right), \tag{3.11}$$

onde f_t é a freqüência transformada (em mels), f_s a freqüência de amostragem (em Hz) e α_{bt} o parâmetro usado para transformar (distorcer) o eixo de freqüências. Quando $0{,}4 \leq \alpha_{bt} \leq 0{,}8$ a distorção do eixo de freqüências da transformação bilinear é similar (próxima) àquela da escala mel. Nas simulações utilizou-se $\alpha_{bt} = 0{,}6$. Os coeficientes cepstrais utilizados na transformação bilinear foram os calculados via LPC. As equações recursivas utilizadas são as dadas por Picone (1993).

Independentemente do algoritmo utilizado, calcularam-se 13 coeficientes mel-cepstrais por quadro. O primeiro coeficiente não foi utilizado por estar relacionado com a potência do sinal.

- Coeficientes mel-cepstrais com subtração da média espectral

Nestes testes, procedeu-se à subtração da média espectral dos coeficientes mel-cepstrais calculados via DCT (Davis & Mermelstein), antes de sua utilização como parâmetros de entrada da rede neural. A forma de realizar esta subtração, é estimar a média de cada um dos coeficientes espectrais ao longo da palavra e subtraí-los dos coeficientes originais correspondentes.

- Delta-Cepstrais e Delta-Mel-Cepstrais

Com a intenção de caraterizar melhor as variações temporais de um sinal, podem adicionar-se ao modelo do sinal as derivadas com respeito ao tempo dos coeficientes originais. Três aproximações usuais para estas derivadas são dadas por:

$$\dot{s}(n) \equiv \frac{d}{dt} s(n) \approx s(n) - s(n-1). \tag{3.12a}$$

$$\dot{s}(n) \equiv \frac{d}{dt} s(n) \approx s(n+1) - s(n). \tag{3.12b}$$

$$\dot{s}(n) \equiv \frac{d}{dt} s(n) \approx \sum_{m=-N}^{N} m.\, s(n+m) \ . \tag{3.13}$$

As primeiras duas equações, 3.12a e 3.12b, são conhecidas como *diferenças para trás* e *para frente* respectivamente. A terceira equação (Eq. 3.13), representa um filtro de fase linear que aproxima um diferenciador ideal usando as N amostras precedentes e as N amostras posteriores à amostra corrente.

O sinal de saída deste processo de diferenciação denomina-se *parâmetro delta*. No trabalho presente os parâmetros delta foram obtidos com a seguinte equação (Rabiner, 1989):

$$d_q(k) = g. \sum_{m=-r}^{r} m.\, coef_{q+m}(k) \ , \qquad k = 1, 2, \ldots, P. \tag{3.14}$$

onde d é o coeficiente delta calculado, k o índice do coeficiente, P o número de coeficientes delta a calcular; q indica o quadro onde o coeficiente está sendo calculado; r define o número de coeficientes (quadros) anteriores e posteriores usados no cálculo dos deltas; *coef* é o tipo de coeficiente empregado (por exemplo *cepstrais* para o cálculo dos *delta-cepstrais*), e g é uma constante de valor

empírico que tenta igualar a variância dos coeficientes originais com a dos coeficientes delta obtidos.

Neste trabalho foram calculados, por quadro, 12 coeficientes *delta-cepstrais*, 12 *delta-mel-cepstrais*, o módulo do vetor de delta-mel-cepstrais e o *delta-mel-módulo*, correspondente ao delta do módulo do vetor mel. Para isto utilizaram-se os 3 quadros anteriores e os 3 posteriores ao quadro em questão ($r = 3$).

- Perfil de energia

O cálculo do perfil de energia foi realizado após fazer a pré-ênfase do sinal de fala. Calculou-se a energia de cada quadro de análise, realizando-se a somatória do valor absoluto das amostras pertencentes ao quadro, elevadas ao quadrado.

$$energia(m) = \sum_{n=-\infty}^{\infty} \left| x(n).w(m-n) \right|^2 = \sum_{n=m-N+1}^{m} \left| x(n).w(m-n) \right|^2 . \qquad (3.15)$$

A energia dos quadros foi posteriormente normalizada com a energia máxima da palavra. Cada quadro de análise gerou 1 coeficiente *energia*.

- Delta-energia

Aplicou-se a Eq. 3.14 sobre a energia calculada no item anterior, obtendo-se 1 coeficiente *delta-energia* por quadro de análise.

- Sonoridade

O parâmetro *sonoridade* nos diz se um quadro do sinal é sonoro ou não-sonoro. Um quadro do sinal é sonoro se nesse quadro o sinal é periódico. Para verificar a sonoridade ou não-sonoridade de um quadro, verifica-se se existe periodicidade no sinal de fala. Não interessa *o valor* da freqüência fundamental do quadro sob análise, senão simplesmente saber se esta freqüência fundamental existe ou não. Para o cálculo do parâmetro sonoridade, utilizou-se uma modificação do algoritmo para cálculo de pitch de Kurt Shäfer-Vincent (1983). Cada quadro de análise gera 1 coeficiente sonoridade, indicando a sonoridade ou não do quadro.

Nos próximos capítulos serão descritas as simulações realizadas ao longo deste trabalho e os resultados obtidos. Explicar-se-ão as técnicas propostas de dizimação e interpolação de quadros para o caso de realizar-se análise com quadros de duração fixa e análise síncrona com o pitch, e mostrar-se-á como estas técnicas melhoram os índices de reconhecimento conseguidos com os métodos de análise tradicionais. Também será mostrado como a adaptação do sistema às características espectrais da voz do falante (locutor), melhora os índices de reconhecimento obtidos.

Capítulo 4

Sistemas Implementados

Os sistemas implementados neste trabalho fazem um reconhecimento *estático* da palavra falada. Por reconhecimento estático entende-se aquele onde a rede neural é alimentada *simultaneamente* com *todos* os vetores de parâmetros resultantes da análise do sinal de fala. Neste tipo de reconhecimento o número de entradas da rede neural fica proporcional ao número de quadros de análise utilizado e ao número de parâmetros espectrais e temporais calculados em cada quadro. O sistema toma uma única decisão para decidir qual foi a palavra falada.

Um outro tipo de reconhecimento possível é o *dinâmico*, onde um pequeno quadro de análise vai sendo deslocado sobre o sinal de voz e o resultado desta análise vai sendo apresentado à rede neural. Esta toma decisões locais para, no final, fazer uma decisão global da palavra falada. Neste caso, o número de entradas da rede neural é proporcional ao número de parâmetros calculados no quadro de análise mencionado. Exemplo de uma rede que faz reconhecimento dinâmico é a TDNN (*Time-Delay Neural Network*, Waibel et al., 1989).

Dado que em reconhecimento estático o número de entradas da rede neural é diretamente proporcional ao número de quadros de análise utilizado, é necessário implementar algum mecanismo que mantenha constante o número de entradas da rede independentemente da duração das palavras. Foi visto no Cap. 3, Seção 3.1, que formas de se conseguir isto eram:

1. ***Analisar todas as palavras com igual número de quadros.***
2. ***Analisar as palavras com quadros de comprimento fixo*** e depois:
   - **a.** colocar o número de entradas da rede suficientemente grande de forma que caiba a palavra mais longa e para as palavras mais curtas completar as entradas vazias da rede com zeros;
   - **b.** dizimar ou interpolar o número de quadros de análise de cada palavra de forma que todas fiquem com igual número de quadros.

Na opção 1 (análise com igual número de quadros em todas as palavras), deve-se esperar o término da locução da palavra para poder começar a analisar a mesma. Na opção 2 (análise com quadros de comprimento fixo), a palavra pode ser analisada à medida em que é falada.

Nas simulações realizadas implementaram-se as opções mencionadas acima e para o caso de dizimar/interpolar quadros pesquisou-se tanto a análise com quadros de comprimento fixo, como a análise *síncrona* com o pitch. Na análise síncrona com o pitch, as janelas de análise estão centradas nas marcas de pitch, e sua duração é de dois períodos de pitch. Este tipo de análise é inédito em reconhecimento de fala e, como será visto, produziu excelentes resultados.

Nas próximas seções serão apresentados os resultados dos sistemas simulados. A seqüência de apresentação será a seguinte:

1. Uso de perceptrons multicamadas. Análise com igual número de quadros para todas as palavras e análise com quadros de comprimento fixo. Definição dos melhores parâmetros espectrais e temporais a utilizar para discriminar as palavras a reconhecer. Testes com redes de 1 e 2 camadas escondidas. Desempenho das redes com sinais ruidosos e sem ruído. Especificação da rede que produziu os melhores resultados.
2. Uso de redes de Kohonen, Kohonen-Grossberg e LVQ. Análise com igual número de quadros para todas as palavras. Mapas de Kohonen lineares e quadrados. Desempenho dos sistemas com sinais ruidosos e sem ruído.
3. Uso de quantização vetorial, utilizando codebooks de 16, 32, 64, 128 e 256 vetores em perceptrons multicamadas. Uso dos vetores do codebook como entradas da rede neural. Desempenho dos sistemas com sinais sem ruído.
4. Uso das técnicas Trace Segmentation e Individual Trace Segmentation, em perceptrons multicamadas e redes LVQ. Análise com quadros de comprimento fixo recalculando os parâmetros a cada 4 ms e a cada 2 ms. Desempenho dos sistemas com sinais sem ruído.
5. Uso de dizimação e interpolação de quadros em perceptrons multicamadas utilizando:

   5.a Análise com quadros de comprimento fixo.

   5.b Análise síncrona com o pitch (quadros de duração variável).

   Critérios usados na dizimação e interpolação. Número de regiões de dizimação e interpolação. Desempenho dos sistemas com sinais ruidosos e sem ruído.

Em todos os casos o desempenho dos sistemas foi medido calculando-se a taxa de erro com a Eq. 1.1 do Cap. 1.

4.1 Uso de Perceptrons Multicamadas - Determinação dos melhores parâmetros espectrais e temporais a utilizar

As simulações desta seção tiveram dois objetivos principais. O primeiro foi definir a rede backpropagation (perceptron multicamadas treinado com o algoritmo backpropagation), que melhor se ajustava às nossas necessidades (número de camadas escondidas, número de neurônios de cada camada, função de ativação dos neurônios, valores da taxa de aprendizado e do fator momento no algoritmo de treinamento, etc.). Com isto poderiam se limitar as configurações a utilizar nas simulações seguintes. O segundo objetivo foi determinar que tipo de análise era mais vantajoso usar –se quadros de comprimento fixo ou igual número de quadros para todas as palavras– e quais parâmetros espectrais e temporais melhor discriminavam as palavras a reconhecer.

Em todos os casos os resultados apresentados correspondem à média dos testes realizados revezando os conjuntos de treinamento e teste como explicado no Cap. 3, Seção 3.1. O critério utilizado para parar o treinamento das redes foi o seguinte: após ter-se atingido um erro de 1,0 a 1,5% com o conjunto de treinamento, realizou-se validação cruzada, i.e., corrigiu-se a rede com o conjunto de treinamento mas testou-se-a com o conjunto de teste, suspendendo o treinamento quando o erro com o conjunto de teste começou a aumentar.

Os conjuntos de treinamento foram construídos colocando as palavras em ordem semi-aleatória. Por ordem semi-aleatória entenda-se que impediu-se a repetição da locução de uma palavra antes que o vocabulário completo fosse apresentado uma vez à rede. A ordem de apresentação das palavras foi a mesma em todas as épocas. Se bem que aqui apresentar-se-ão os resultados de simulações feitas com vozes masculinas e femininas misturadas, também se testaram sistemas com vozes unicamente masculinas ou unicamente femininas. Nesses casos os erros de reconhecimento foram menores que com sistemas mistos chegando, para o vocabulário de dígitos, a cair pela metade.

Os sistemas foram treinados e testados com sinais sem ruído e com sinais ruidosos. Os sinais ruidosos tiveram SNR de 20 dB e 10 dB.

4.1.1 Análise com igual número de quadros em todas as palavras

Quando se analisam todas as palavras com igual número de quadros, quadros de diferentes elocuções de uma mesma palavra tendem a corresponder aos mesmos sons. Neste trabalho foi testado o uso de 40 a 80 quadros de análise por palavra com saltos de 5 quadros (40 quadros, 45 quadros, 50 quadros, etc.). Os quadros foram ponderados com janelas de Hamming, usando-se 1/3 de superposição entre quadros consecutivos. Palavras mais longas tiveram quadros mais longos. Com a base de dados utilizada e com 40 quadros de análise por palavra, a duração dos quadros variou entre 10,8 ms e 42 ms. Com 80 quadros de análise por palavra, a duração dos quadros variou entre 5,4 ms e 21 ms.

Os testes mostraram que o uso de 40 a 60 quadros por palavra é o que dá os melhores resultados e que estes são comparáveis entre si. Visto que quanto menor for o número de quadros de análise menor é a carga computacional do sistema, optou-se por utilizar 40 quadros de análise nos testes subsequentes. Os resultados aqui apresentados correspondem a este número de quadros.

Na primeira fase foram testados os parâmetros *LPC* e de *reflexão* obtidos com o algoritmo recursivo de Levinson-Durbin, os parâmetros *cepstrais* obtidos a partir dos LPC, os parâmetros *mel-cepstrais* obtidos com o algoritmo de Davis & Mermelstein, os parâmetros *delta-cepstrais* e *delta-mel-cepstrais* obtidos com a Eq. 3.14 do Cap. 3, as combinações *"sonoridade com LPC"*, *"sonoridade com parâmetros de reflexão"*, *"sonoridade com parâmetros cepstrais"*, *"sonoridade com parâmetros mel-cepstrais"*, *"cepstrais com delta-cepstrais"*, *"mel-cepstrais com delta-mel-cepstrais"*, *"mel-cepstrais com delta-cepstrais e delta-mel-cepstrais"* e, finalmente, a combinação de *"mel-cepstrais com energia"* (os algoritmos utilizados para a obtenção destes parâmetros são os apresentados no Cap. 3). Os parâmetros *delta* foram calculados utilizando-se os 3 quadros anteriores e os 3 posteriores ao quadro em questão. Em todos os casos os sinais de fala utilizados foram os originais (SNR superior a 50 dB), pré-enfatizados com um fator de 0,9.

As tabelas e figuras seguintes apresentam os resultados principais. Nelas, a legenda entre parêntesis ao lado de um parâmetro indica o algoritmo utilizado no seu cálculo. Por exemplo "Ceps (LPC)" indica coeficientes cepstrais calculados via LPC. Os valores das taxas de erro são mostrados em função do número de neurônios da camada escondida (**h**). Nas tabelas são mostrados os resultados para todos os parâmetros e combinações de parâmetros mencionados.

Dado que, como será visto, os parâmetros que produziram os melhores resultados foram os cepstrais, os mel-cepstrais e a combinação de mel-cepstrais com energia, nas figuras são mostrados, apenas, os resultados para estes três parâmetros e para os parâmetros LPC (estes últimos para efeito de comparação).

As tabelas e figuras 4.1 a 4.3 correspondem à utilização de uma rede backpropagation de uma camada escondida, com o número de neurônios desta camada variando de 20 a 60 em incrementos de 10. O número de entradas da rede neural foi feito igual ao produto do número de parâmetros a utilizar por quadro de análise, pelo número de quadros (40 neste caso). Com isto o número de entradas da rede variou de 480 a 960. O número de saídas da rede foi feito igual ao número de palavras do vocabulário a reconhecer (10 saídas para os dígitos, e 11 para os comandos de cálculo e de movimento).

Todos os neurônios tiveram uma entrada adicional de valor 1 (bias), com sinapse treinável. A função de ativação utilizada foi a sigmóide binária (função logística), com $\sigma = 1$ (Eq. 2.5, Cap. 2). Os valores de taxa de aprendizagem η e do fator momento α utilizados foram: 0,3 a 0,35 para a taxa de aprendizagem e 0,7 e 0,85 para o fator momento. Estes valores são os que provocaram os menores tempos de convergência da rede. O número de épocas de treinamento ficou em todos os casos menor que 65. A tabela e figura 4.1 corresponde aos dígitos, a 4.2 aos comandos de cálculo e a 4.3 aos comandos de movimento.

Tabela 4.1: Taxas de erro em % para os dígitos, utilizando diferentes parâmetros de entrada e 40 quadros de análise por palavra. Utiliza-se uma rede backpropagation com 1 camada escondida e com o número de neurônios desta camada variando de 20 a 60 (valor de h). Parâmetro "S": sonoridade.

Parâmetros	h = 20	30	40	50	60
LPC	5,31	5,31	5,31	5,63	5,63
Ref	5,63	5,94	6,25	6,25	6,56
Ceps (LPC)	3,44	2,50	2,81	2,81	2,81
Mel (DCT)	2,50	1,88	2,19	2,50	2,50
delta-ceps (dc)	5,94	5,94	5,94	6,25	6,25
delta-mel (dm)	4,38	4,38	4,69	4,69	5,00
LPC+S	4,06	3,75	4,06	4,06	4,69
Ref+S	5,94	5,63	6,25	6,56	6,88
Ceps+S	5,31	4,69	4,38	4,69	5,31
Mel+S	3,13	2,81	2,81	3,13	3,44
Ceps+8dc	4,38	4,06	4,06	4,38	5,00
Mel+8dm	3,13	2,81	2,81	3,13	3,44
Mel+6dm+6dc	3,13	2,50	2,50	2,81	3,13
Mel+energia	2,50	2,19	2,19	2,81	2,81

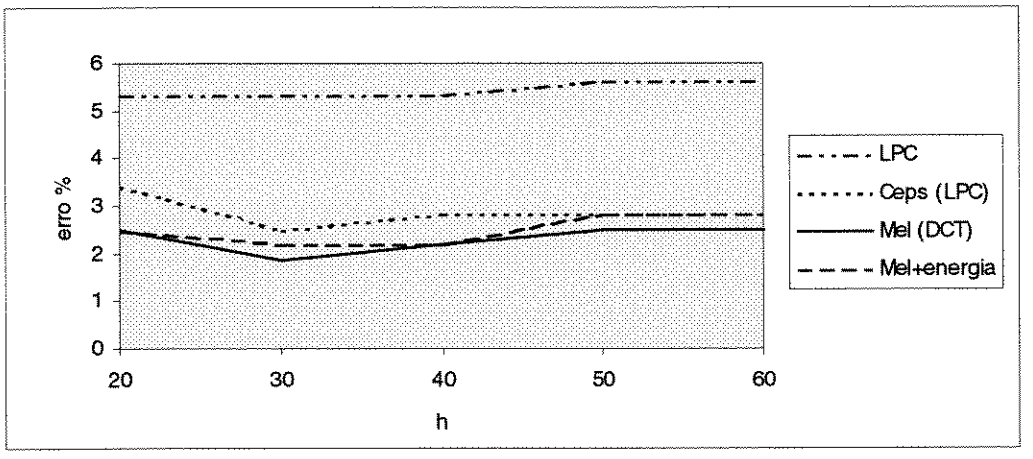


Figura 4.1: Taxas de erro em % para os dígitos, em função do número de neurônios da camada escondida e dos parâmetros de entrada utilizados. Análise com 40 quadros por palavra (origem: Tabela 4.1).

Tabela 4.2: Taxas de erro em % para os comandos de cálculo utilizando diferentes parâmetros de entrada e 40 quadros de análise por palavra. Utiliza-se uma rede backpropagation com 1 camada escondida e com o número de neurônios desta camada variando de 20 a 60 (valor de h).

Parâmetros	h = 20	30	40	50	60
LPC	5,68	5,30	4,92	5,68	5,68
Ref	4,17	3,79	3,79	4,17	4,92
Ceps (LPC)	4,17	3,41	3,03	4,17	4,55
Mel (DCT)	4,17	3,03	3,03	3,03	3,41
Delta-ceps (dc)	6,06	5,68	5,68	6,06	6,44
Delta-mel (dm)	3,79	3,41	3,41	3,79	4,17
LPC+S	4,17	3,41	3,41	3,79	4,17
Ref+S	3,79	3,79	3,79	4,55	4,92
Ceps+S	4,17	3,79	3,79	4,17	4,55
Mel+S	3,03	2,65	2,65	3,03	3,41
Ceps+8dc	4,92	4,17	4,55	4,92	4,92
Mel+8dm	3,03	2,27	2,27	2,65	3,03
Mel+6dm+6dc	3,03	2,27	2,27	2,65	3,03
Mel+energia	2,65	2,65	2,27	2,65	3,03

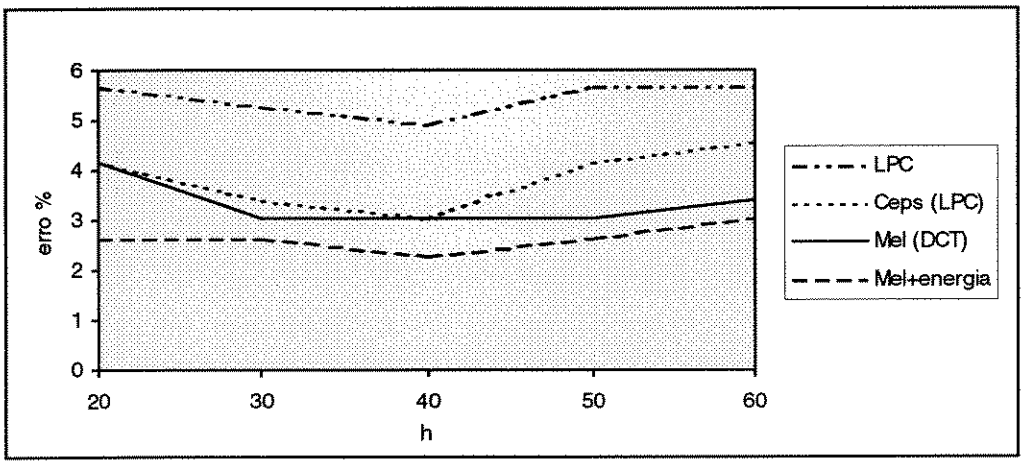


Figura 4.2: Taxas de erro em % para os comandos de cálculo em função do número de neurônios da camada escondida e dos parâmetros de entrada utilizados. Análise com 40 quadros por palavra (origem: Tabela 4.2).

Tabela 4.3: Taxas de erro em % para os comandos de movimento utilizando diferentes parâmetros de entrada e 40 quadros de análise por palavra. Utiliza-se uma rede backpropagation com 1 camada escondida e com o número de neurônios desta camada variando de 20 a 60 (valor de h).

Parâmetros	h = 20	30	40	50	60
LPC	9,85	9,47	9,09	9,47	9,85
Ref	9,09	8,71	8,71	9,09	9,47
Ceps (LPC)	4,17	3,41	4,17	5,30	4,17
Mel (DCT)	4,17	2,65	3,41	3,41	3,41
delta-ceps (dc)	9,85	9,47	8,71	9,47	9,85
delta-mel (dm)	7,58	6,44	6,44	7,20	7,58
LPC+S	11,74	9,09	9,85	11,74	12,12
Ref+S	11,36	8,71	10,61	11,36	11,74
Ceps+S	6,44	5,68	5,68	6,06	6,82
Mel+S	3,03	2,65	3,41	3,41	3,79
Ceps+8dc	4,55	4,55	5,68	6,82	6,82
Mel+8dm	3,41	3,03	3,03	3,41	3,79
Mel+6dm+6dc	4,17	4,17	4,55	4,55	5,68
Mel+energia	3,03	2,65	2,27	2,65	3,41

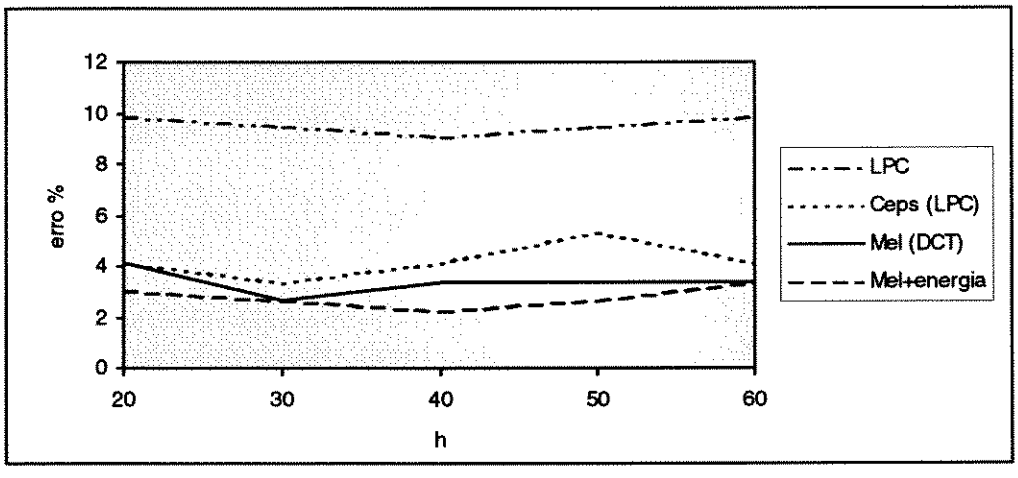


Figura 4.3: Taxas de erro em % para os comandos de movimento em função do número de neurônios da camada escondida e dos parâmetros de entrada utilizados. Análise com 40 quadros por palavra (origem: Tabela 4.3).

Das tabelas acima pode ser visto que, em geral, os parâmetros que apresentaram os melhores resultados foram os mel-cepstrais, os cepstrais e a combinação mel-cepstrais com energia. O uso de coeficientes LPC e de reflexão aumentou a taxa de erro substancialmente. Em alguns casos este aumento superou 100%.

Em relação aos parâmetros delta-cepstrais e delta-mel-cepstrais, verificou-se que quando empregados isoladamente (o que não é o usual), apresentaram um desempenho inferior ao dos parâmetros dos quais provieram e um desempenho similar aos dos coeficientes LPC e de reflexão. Seu uso em combinação com coeficientes cepstrais e mel-cepstrais não melhorou o desempenho dos

sistemas. Uma justificativa para este comportamento é que a informação que os parâmetros delta carregam provavelmente já está presente na rede –mesmo sem usá-los– como combinação linear dos parâmetros originais (isto por realizar-se reconhecimento *estático* da palavra falada). Assim a inclusão de parâmetros delta não incorpora informações novas à rede e os índices de acerto permanecem praticamente inalterados.

Quanto à *sonoridade*, seu uso permitiu, em alguns casos, melhorar os índices de reconhecimento dos parâmetros LPC e de reflexão, mas sem que com isto se superassem os índices de reconhecimento conseguidos com os mel-cepstrais. A combinação de sonoridade com parâmetros cepstrais ou mel-cepstrais prejudicou a capacidade discriminante da rede. Provavelmente isto é devido a erros na decisão sonoro/não-sonoro e ao fato de coexistir em alguns dos quadros tanto trechos sonoros como não-sonoros (em diversas elocuções de uma mesma palavra, isto causa diferentes decisões quanto à sonoridade ou não de um quadro).

Também foi testado o uso isolado do parâmetro *energia* como única entrada à rede neural. Neste caso não foi possível diminuir o erro de treinamento aquém de 10,3%, ficando a taxa de erro da rede maior que 28,7%.

Nas tabelas acima pode verificar-se uma tendência de aumento nas taxas de erro para **h** maior que 40 (número de neurônios da camada escondida). A explicação para este fato é que à medida que o número de sinapses aumenta, é necessária uma seqüência de treinamento maior para se obter uma boa estimativa das mesmas. Dado que aqui foi aumentado **h** mantendo-se inalterado o conjunto de treinamento, as estimativas das sinapses pioraram e as taxas de erro aumentaram. Aumentando-se o conjunto de treinamento, as taxas de erro devem diminuir.

Até aqui foi utilizada como função de ativação dos neurônios a sigmóide binária. Utilizando-se como função de ativação a sigmóide bipolar com $\sigma = 1$ (Eq. 2.6, Cap. 2), obtiveram-se taxas de erro similares às obtidas com a sigmóide binária. Verificou-se, no entanto, em alguns casos, a diminuição dos tempos de treinamento em até 30%. A tangente hiperbólica (Eq. 2.7, Cap. 2), não foi testada devido a ser um caso particular da sigmóide bipolar para $\sigma = 2$.

Quanto aos tempos gastos no treinamento, verificou-se que utilizando coeficientes mel-cepstrais os tempos foram entre 3 e 5 vezes maiores que quando se utilizaram coeficientes cepstrais. A hipótese para explicar este fato é o intervalo de variação dos coeficientes mel-cepstrais. Com os algoritmos e vocabulários aqui utilizados o intervalo de variação dos coeficientes mel-cepstrais é quase 7

vezes maior que o intervalo de variação dos coeficientes cepstrais. Assim, ao se utilizar coeficientes mel-cepstrais cai-se na região da sigmóide onde a derivada é praticamente nula (região horizontal da curva). Dado que o fator de correção do algoritmo backpropagation é função do valor desta derivada, se a derivada é pequena a correção resulta pequena e a rede demora mais tempo para convergir. Se a hipótese é correta, modificando a pendente da função de ativação dos neurônios (valor de σ), para cair na região de máxima derivada da curva, o problema deve resolver-se. Repetindo as Eqs. 2.5a e 2.5b (função sigmóide binária e sua derivada), tem-se:

$$f(x) = \frac{1}{1+\exp(-\sigma\, x)} \quad ; \quad f'(x) = \sigma\ f(x)[1-f(x)] \tag{2.5}$$

Fausset (1994), dá a seguinte equação para calcular o σ necessário quando os parâmetros de entrada variam entre x_{min} a x_{max}, com $x_{min} = -x_{max}$ (isto é, centrados em zero):

$$\sigma = \frac{\ln(3)}{x_{max}} \tag{4.1}$$

Para os vocabulários aqui utilizados, o intervalo de variação dos coeficientes mel-cepstrais calculados com o algoritmo de Davis & Mermelstein foi de −19,35 a 20,59. Escolhendo-se $x_{max} = 21$ obtém-se $\sigma = 0{,}052$. Utilizando esta pendente na sigmóide binária os tempos de treinamento caíram a valores similares aos obtidos com os coeficientes cepstrais, confirmando a hipótese sugerida. Os valores das taxas de erro permaneceram os mesmos.

Mantendo a pendente da função de ativação inalterada e dividindo-se os coeficientes mel-cepstrais por 7 (número de vezes que o intervalo de variação dos mel-cepstrais é maior que o dos cepstrais), os tempos de treinamento também diminuem a valores similares ao dos cepstrais, mantendo as taxas de erro inalteradas. Observe que, se ao invés de se dividir as entradas da rede por 7, se faz a divisão pela valor da máxima entrada, o procedimento equivale a *normalizar* as entradas da rede neural. A opção de normalizar as entradas da rede não foi implementada, por terem sido obtidos resultados satisfatórios com os procedimentos anteriores.

Nas simulações realizadas as funções sigmóide binária e sigmóide bipolar produziram praticamente as mesmas taxas de erro, influenciando, unicamente, nos tempos de treinamento gastos. Ao utilizar coeficientes mel-cepstrais, o uso de uma inclinação adequada na função sigmóide diminuiu os tempos de treinamento, em média, a uma quarta parte dos originais. Dado que os coeficientes

cepstrais tiveram um intervalo de variação maior que 1 (com a base de dados empregada e calculados via LPC o intervalo de variação dos coeficientes cepstrais foi de –2,77 a 2,52), deveria utilizar-se com eles uma pendente $\sigma = 0{,}39$ na função de ativação (Eq. 4.1 com $x_{max} = 2{,}8$). Isto não foi feito, já que os tempos de treinamento com $\sigma = 1$ foram razoáveis.

A tabela e figura seguintes, 4.4, mostram as taxas de erro obtidas ao usar parâmetros cepstrais, mel-cepstrais e a combinação "mel-cepstrais com energia" no vocabulário completo (união dos três vocabulários anteriores). Os outros parâmetros não foram utilizados devido aos baixos índices de reconhecimento conseguidos nos vocabulários originais. A rede utilizada foi uma backpropagation com uma camada escondida contendo de 20 a 60 neurônios. A função de ativação utilizada foi a sigmóide binária (como nas Tabelas 4.1 a 4.3).

Tabela 4.4: Taxas de erro em % para o vocabulário completo (32 palavras), utilizando parâmetros cepstrais, mel-cepstrais e energia e 40 quadros de análise por palavra. Utiliza-se uma rede backpropagation com 1 camada escondida e com o número de neurônios desta camada variando de 20 a 60 (valor de h).

Parâmetros	h = 20	30	40	50	60
Ceps (LCP)	11,59	8,72	8,72	8,59	7,42
Mel (DCT)	14,32	6,64	6,51	6,51	6,51
Mel+energia	11,20	8,33	8,07	6,77	6,81

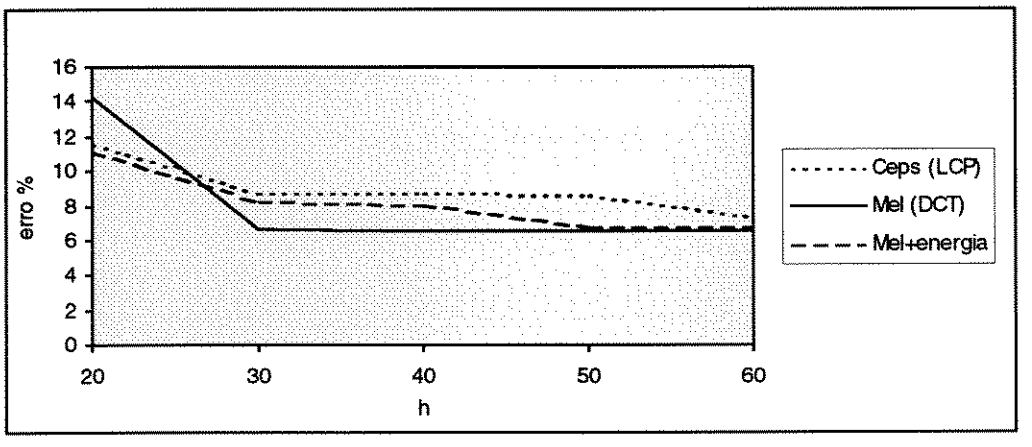


Figura 4.4: Taxas de erro em % para o vocabulário completo em função do número de neurônios da camada escondida e dos parâmetros de entrada utilizados. Análise com 40 quadros por palavra (origem: Tabela 4.4).

Como ocorreu com os três vocabulários anteriores, os parâmetros que produziram os melhores resultados foram os mel-cepstrais e a combinação de mel-cepstrais com energia. Observa-se aqui, porém, que o aumento do número de neurônios escondidos além de 40, não provoca um aumento nas taxas de erro do sistema. Isto se explica considerando que para o vocabulário completo o con-

junto de treinamento é bem maior que para os vocabulários individuais. Portanto, o efeito de perda de precisão na estimativa das sinapses ocorre a partir de um número maior de neurônios escondidos.

A Tabela 4.5 é um resumo das tabelas anteriores onde se apresentam, apenas, os resultados obtidos com uma rede backpropagation com 30 neurônios na camada escondida (pode ser visto nas tabelas anteriores que nem sempre este número de neurônios produziu os melhores resultados). Na Fig. 4.5 a coluna mais à esquerda de cada grupo corresponde aos dígitos, a seguinte aos comandos de cálculo, a seguinte aos comandos de movimento e a seguinte ao vocabulário completo.

Tabela 4.5: Taxas de erro em % para os 4 vocabulários utilizando diferentes parâmetros de entrada e 40 quadros de análise por palavra. Utiliza-se uma rede backpropagation com 1 camada escondida de 30 neurônios.

Parâmetros	Dígitos (D)	Cálculo (C)	Movimento (M)	D+C+M
LPC	5,31	5,30	9,47	----
Ref	5,94	3,79	8,71	----
Ceps (LPC)	2,50	3,41	3,41	8,72
Mel (DCT)	1,88	3,03	2,65	6,64
delta-ceps (dc)	5,94	5,68	9,47	----
delta-mel (dm)	4,38	3,41	6,44	----
LPC+S	3,75	3,41	9,09	----
Ref+S	5,63	3,79	8,71	----
Ceps+S	4,69	3,79	5,68	----
Mel+S	2,81	2,65	2,65	----
Ceps+8dc	4,06	4,17	4,55	----
Mel+8dm	2,81	2,27	3,03	----
Mel+6dm+6dc	2,50	2,27	4,17	----
Mel+energia	2,19	2,65	2,65	8,33

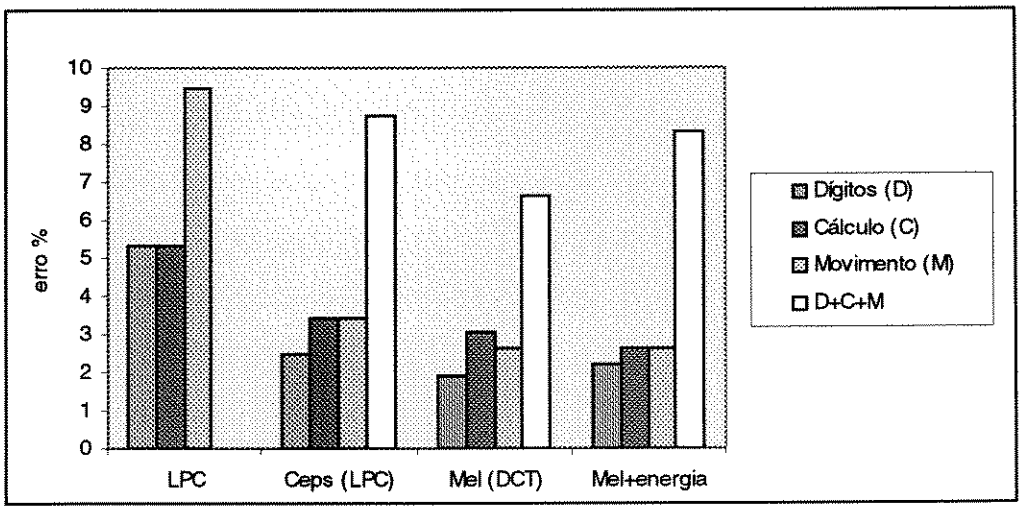


Figura 4.5: Taxas de erro em % para os 4 vocabulários em função dos parâmetros de entrada utilizados. Análise com 40 quadros por palavra e 30 neurônios na camada escondida (origem: Tabela 4.5).

Teste de coeficientes cepstrais e mel-cepstrais calculados com algoritmos alternativos

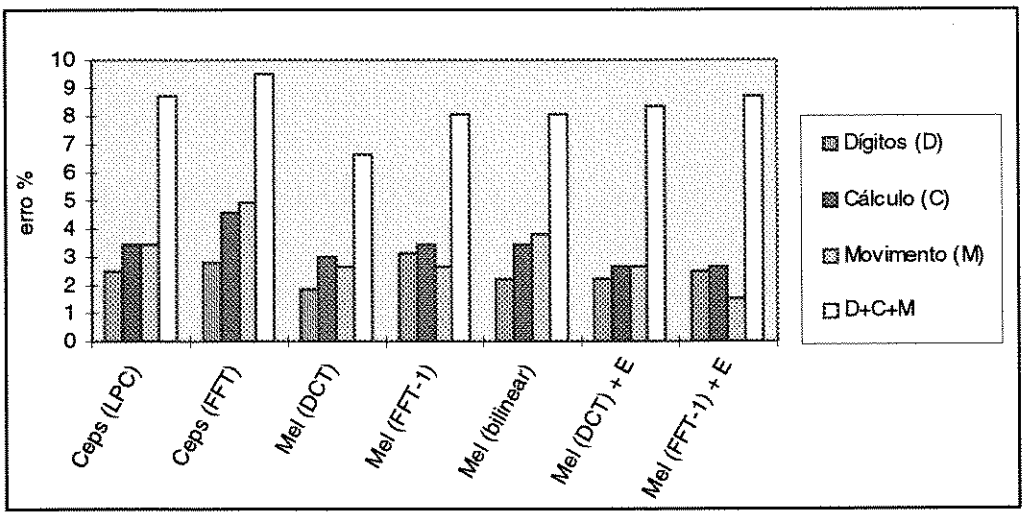
No item anterior foi visto que os melhores parâmetros discriminantes foram os cepstrais, os mel-cepstrais, e a combinação de mel-cepstrais com energia. Nesta segunda fase serão testados os parâmetros cepstrais e mel-cepstrais obtidos com os algoritmos alternativos apresentados no Cap. 3. Os coeficientes cepstrais serão calculados via FFT ao invés de via LPC. Os coeficientes mel-cepstrais serão calculados com 2 algoritmos alternativos: o sugerido por Deller et al. (1993) que usa FFT^{-1} ao invés de DCT, e aquele que calcula os mel-cepstrais a partir dos coeficientes cepstrais utilizando uma transformação bilinear. Neste último caso os coeficientes cepstrais a empregar serão os calculados via LPC.

A Tabela 4.6 mostra o resultado de utilizar os coeficientes obtidos com os algoritmos alternativos nos 4 vocabulários sob teste. A tabela inclui, para efeito de comparação, os resultados obtidos com os coeficientes cepstrais calculados via LPC e com os coeficientes mel-cepstrais calculados via DCT. A rede utilizada é uma backpropagation com 1 camada escondida contendo 30 ou 50 neurônios. A função de ativação utilizada é a sigmóide binária.

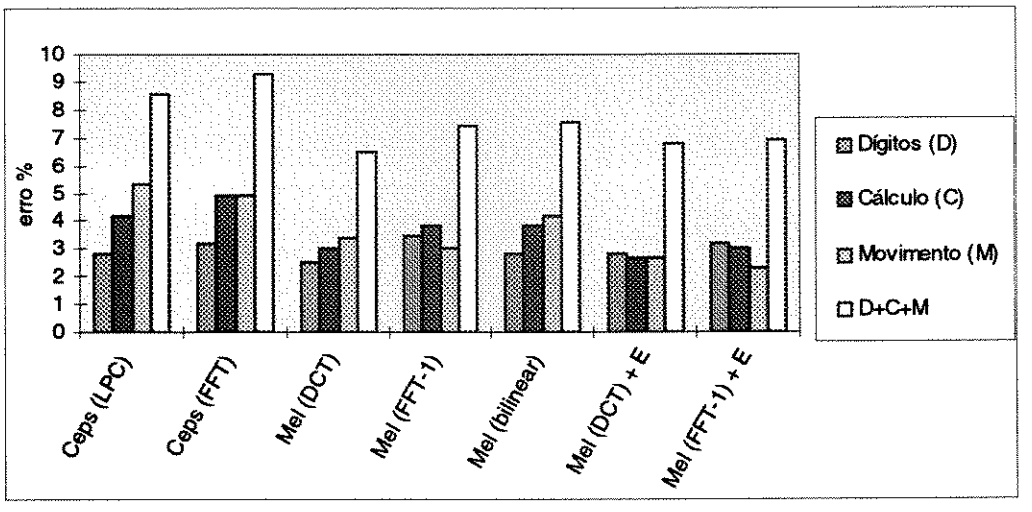
A figura 4.6 (a) plota os valores destes erros quando a camada escondida tem 30 neurônios e a figura 4.6 (b) quando a camada escondida tem 50 neurônios. Como antes, a coluna mais à esquerda de cada grupo corresponde aos dígitos, a seguinte aos comandos de cálculo, a seguinte aos comandos de movimento e a seguinte ao vocabulário completo.

Tabela 4.6: Taxas de erro em % para os 4 vocabulários utilizando parâmetros cepstrais e mel-cepstrais obtidos com diferentes algoritmos e a combinação mel-cepstrais com energia (E). Análise com 40 quadros por palavra. Utiliza-se uma rede backpropagation com 1 camada escondida contendo 30 ou 50 neurônios (valor de h).

	Dígitos (D)		Cálculo (C)		Movimento (M)		D+C+M	
Parâmetros	**h = 30**	**50**	**30**	**50**	**30**	**50**	**30**	**50**
Ceps (LPC)	2,50	2,81	3,41	4,17	3,41	5,30	8,72	8,59
Ceps (FFT)	2,81	3,13	4,55	4,92	4,92	4,92	9,51	9,25
Mel (DCT)	1,88	2,50	3,03	3,03	2,65	3,41	6,64	6,51
Mel (FFT^{-1})	3,13	3,44	3,41	3,79	2,65	3,03	8,07	7,42
Mel (bilinear)	2,19	2,81	3,41	3,79	3,79	4,17	8,07	7,55
Mel (DCT) + E	2,19	2,81	2,65	2,65	2,65	2,65	8,33	6,77
Mel (FFT^{-1}) + E	2,50	3,13	2,65	3,03	1,52	2,27	8,72	6,90



(a)



(b)

Figura 4.6: Taxas de erro em % para os 4 vocabulários utilizando parâmetros cepstrais e mel-cepstrais obtidos com diferentes algoritmos, e a combinação de mel-cepstrais com energia. (a) Usando uma camada escondida com 30 neurônios; (b) usando uma camada escondida com 50 neurônios. Análise com 40 quadros por palavra (origem: Tabela 4.6).

Da análise das tabelas e gráficos anteriores podem-se tirar as seguintes conclusões: os parâmetros *cepstrais* calculados via LPC são levemente superiores aos obtidos via FFT. De igual modo, os parâmetros *mel-cepstrais* calculados com o algoritmo de Davis & Mermelstein, são melhores (causam menos erros de reconhecimento), que os calculados pelos outros dois métodos. Isto pode creditar-se ao fato de que no algoritmo de Davis & Mermelstein é utilizada a DCT (Discrete Cosine Transform), ao invés da FFT^{-1} usada no algoritmo de Deller et al. Uma propriedade que a DCT

possui é a de ordenar estatisticamente os coeficientes obtidos colocando os de maior variância em primeiro lugar. Assim, os 12 primeiros coeficientes mel-cepstrais obtidos com a DCT provavelmente carreguem mais informação discriminante que os 12 primeiros coeficientes obtidos com a FFT^{-1} ou com a transformação bilinear. Isto explicaria a melhor capacidade discriminante dos mel-cepstrais calculados com a DCT.

Além dos melhores índices de reconhecimento que os coeficientes mel-cepstrais de Davis & Mermelstein proporcionam, existe um efeito secundário que diz respeito ao tempo de convergência da rede neural durante o treinamento. Verifica-se que o uso dos coeficientes mel-cepstrais obtidos com o algoritmo sugerido por Deller et al., provoca uma convergência muito lenta da rede neural, sendo necessárias, em média, 6 vezes mais épocas de treinamento que no caso de se utilizar os coeficientes mel-cepstrais de Davis & Mermelstein. Isto não decorre da diferente faixa de variação desses parâmetros, já que em ambos os casos modificou-se a pendente da função de ativação dos neurônios com a Eq. 4.1. Finalmente, a combinação de mel-cepstrais com energia melhorou os índices de reconhecimento da rede em vários dos casos.

Quanto aos tempos de cálculo de cada um dos parâmetros, os coeficientes cepstrais calculados via LPC são os mais rápidos de obter. Seu cálculo via FFT demora, em média, 4 vezes mais tempo. Para os coeficientes mel-cepstrais, tem-se que os mais rápidos de calcular são os obtidos via transformação bilinear (inclui-se aqui o tempo gasto para calcular os coeficientes cepstrais via LPC a partir dos quais são obtidos os mel-cepstrais). No entanto, estes coeficientes mel-cepstrais provocam maiores taxas de erro que os calculados via FFT^{-1}. Os coeficientes mel-cepstrais calculados via DCT demoram menos em ser obtidos que os calculados via FFT^{-1} e, adicionalmente, permitem obter taxas de acerto maiores.

Do exposto nos parágrafos acima temos que se o tempo de cálculo dos parâmetros não é um fator decisivo na escolha dos mesmos, então os coeficientes mel-cepstrais calculados com o algoritmo de Davis & Mermelstein são a melhor opção. Se, pelo contrário, o tempo de cálculo é um fator limitativo, é aconselhável usar os parâmetros cepstrais calculados via LPC. Deve-se destacar aqui, que o pré-processamento do sinal de fala, que inclui a pré-ênfase, janelamento e cálculo dos parâmetros de entrada à rede neural, é responsável por mais de 95% do tempo total gasto em reconhecer a palavra (isto, utilizando perceptrons multicamadas). Portanto, o pré-processamento é o fator principal de demora, e o que consome mais recursos computacionais.

A tabela seguinte, 4.7, mostra os intervalos de variação dos coeficientes cepstrais e mel cepstrais calculados com os diferentes algoritmos mencionados, na base de dados utilizada (os valores mostrados correspondem à utilização de 40 quadros de análise por palavra).

Tabela 4.7: Intervalo de variação dos coeficientes cepstrais e mel-cepstrais com os diferentes algoritmos de cálculo e com 40 quadros de análise por palavra.

	Ceps (LPC)	Ceps (FFT)	Mel (DCT)	Mel (FFT^{-1})	Mel (bilinear)
x_{min}	-2,77	-0,47	-19,35	-16,72	-2,01
x_{max}	2,52	0,45	20,59	16,29	2,83

Uso de coeficientes mel-cepstrais com subtração da média espectral

Nestes testes, procedeu-se à subtração da média espectral dos coeficientes mel-cepstrais calculados via DCT (Davis & Mermelstein), antes de sua utilização como parâmetros de entrada da rede neural. A justificativa para realizar esta subtração, segundo Junqua & Haton (1996), é que a média do espectro de um sinal de fala representa a distorção do canal; portanto, subtraindo-se a média do espectro do sinal de fala remove-se esta distorção. Em HMM (modelos ocultos de Markov), este procedimento permitiu melhorar os índices de acerto dos sistemas de reconhecimento sob teste em até 2% (Martins, 1997).

A forma de realizar a subtração da média espectral, é estimar a média de cada um dos coeficientes espectrais ao longo da palavra e subtraí-los dos coeficientes originais correspondentes.

A subtração espectral foi testada com coeficientes mel-cepstrais e redes backpropagation de uma camada escondida, com 20, 30 e 40 neurônios nesta camada. As taxas de erro obtidas com estes coeficientes foram as mesmas obtidas sem realizar-se a subtração espectral (coeficientes originais). Assim, dado que a subtração espectral não melhorou os índices de acerto, não foi aplicada nas simulações seguintes.

4.1.2 Análise com quadros de comprimento fixo

O objetivo aqui foi determinar se a análise com quadros de comprimento fixo era melhor, sob o ponto de vista de índice de acertos da rede, que a análise com igual número de quadros para todas

as palavras (seção anterior). Para tanto, as redes utilizadas e os parâmetros testados foram os mesmos da seção anterior.

A análise com quadros de comprimento fixo permite que o sinal de fala seja analisado à medida em que a palavra é falada. Em nosso caso utilizamos quadros de 30 ms a cada 20 ms. Isto implica na obtenção de 50 quadros de análise por segundo. Como as palavras têm diferentes durações, o número de quadros de análise a obter depende da duração da palavra. Para manter constante o número de entradas da rede neural, colocou-se este número suficientemente grande de forma que coubesse a palavra mais longa, e para as palavras mais curtas completaram-se as entradas vazias da rede com zeros. Os quadros foram ponderados com janela de Hamming.

A rede neural utilizada foi uma backpropagation, com 1 camada escondida contendo de 20 a 60 neurônios em incrementos de 10. O número de saídas da rede foi feito igual ao número de palavras do vocabulário a reconhecer (10, 11 ou 32). O número de entradas variou de 696 a 1752, em função do vocabulário e dos parâmetros de entrada utilizados. A função de ativação usada foi a sigmóide binária com $\sigma = 1$ (Eq. 2.5, Cap. 2). Os valores de taxa de aprendizagem η e de fator momento α utilizados foram: 0,3 a 0,35 para a taxa de aprendizagem, e 0,7 e 0,85 para o fator momento.

Como entradas da rede neural testaram-se os parâmetros e combinações de parâmetros testados na Seção 4.1.1 (análise com igual número de quadros para todas as palavras). Os parâmetros calculados foram: *LPC* e de *reflexão*, com o método recursivo de Levinson-Durbin, *cepstrais* calculados via LPC, *mel-cepstrais* calculados via DCT (algoritmo de Davis & Mermelstein), parâmetros *delta* calculados com a Eq. 3.14 do Cap.3, utilizando-se os 3 quadros anteriores e os 3 posteriores ao quadro em questão, *energia* calculada com a Eq. 3.15 do Cap.3, e *sonoridade* determinada com o detetor de pitch baseado no *paper* de Kurt Schäfer-Vincent (1983). Quando se quer analisar o sinal de fala à medida em que este chega sem introduzir atrasos, o cálculo dos parâmetros delta deve levar em conta, unicamente, a informação dos quadros anteriores (diferenças para trás). Os sinais de fala utilizados foram os originais (SNR superior a 50 dB), pré-enfatizados com um fator de 0,9.

Em todos os casos o número de épocas de treinamento ficou menor que 65 (similar às simulações da seção anterior). As tabelas e figuras seguintes mostram os resultados obtidos desta vez. As tabelas mostram os resultados para todos os parâmetros e combinações de parâ-

metros testados, enquanto que as figuras mostram os resultados, apenas, para os parâmetros cepstrais, mel-cepstrais, a combinação de mel-cepstrais com energia, e para os parâmetros LPC (estes últimos para efeito de comparação).

A tabela e figura 4.8 corresponde aos dígitos, a 4.9 aos comandos de cálculo, a 4.10 aos comandos de movimento e a 4.11 ao vocabulário completo.

Tabela 4.8: Taxas de erro em % para os dígitos, utilizando diferentes parâmetros de entrada e análise com quadros de 30 ms recalculados a cada 20 ms. Utiliza-se uma rede backpropagation com 1 camada escondida e com o número de neurônios desta camada variando de 20 a 60 (valor de h).

Parâmetros	h = 20	30	40	50	60
LPC	6,88	6,56	6,56	6,88	6,88
Ref	9,38	9,38	9,38	9,69	9,69
Ceps (LPC)	4,06	3,44	3,44	3,75	4,06
Mel (DCT)	2,50	2,19	2,19	2,50	2,50
delta-ceps (dc)	8,75	8,13	8,44	8,44	9,06
delta-mel (dm)	6,88	6,56	6,25	6,56	6,88
LPC+S	8,44	8,13	8,13	8,75	9,06
Ref+S	10,94	10,62	10,31	10,94	11,25
Ceps+S	3,75	3,44	3,13	3,44	4,06
Mel+S	2,19	1,88	1,88	2,19	2,50
Ceps+8dc	3,44	3,13	3,13	3,44	3,44
Mel+8dm	3,44	3,13	2,81	3,13	3,44
Mel+6dm+6dc	3,13	3,13	2,81	3,13	3,44
Mel+energia	2,19	1,88	1,88	2,19	2,19

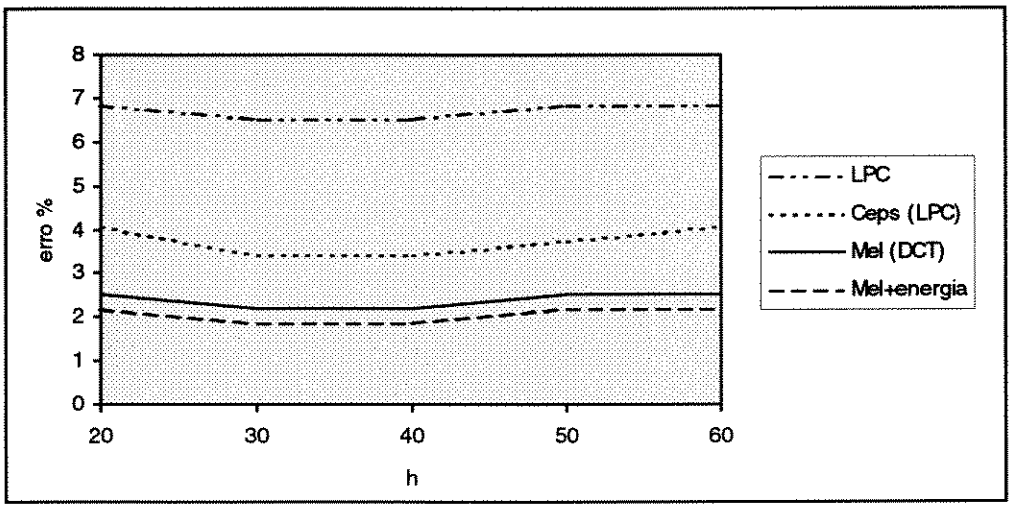


Figura 4.8: Taxas de erro em % para os dígitos em função do número de neurônios da camada escondida e dos parâmetros de entrada utilizados. Análise com quadros de 30 ms recalculados a cada 20 ms (origem: Tabela 4.8).

Tabela 4.9: Taxas de erro em % para os comandos de cálculo utilizando diferentes parâmetros de entrada e análise com quadros de 30 ms recalculados a cada 20 ms. Utiliza-se uma rede backpropagation com 1 camada escondida e com o número de neurônios desta camada variando de 20 a 60 (valor de h).

Parâmetros	h = 20	30	40	50	60
LPC	5,30	4,92	4,17	4,92	4,92
Ref	4,55	4,17	4,17	4,17	4,55
Ceps (LPC)	4,17	3,41	3,03	3,41	3,79
Mel (DCT)	3,79	3,03	3,03	3,41	3,79
delta-ceps (dc)	5,30	4,92	4,92	5,30	5,30
delta-mel (dm)	4,55	4,17	4,17	4,17	4,92
LPC+S	3,79	3,41	3,03	3,41	3,41
Ref+S	4,55	4,17	4,17	4,55	4,55
Ceps+S	5,30	4,92	4,55	4,92	5,30
Mel+S	4,17	4,17	3,41	4,17	4,17
Ceps+8dc	5,68	5,30	4,92	5,30	5,68
Mel+8dm	4,17	3,79	3,79	4,17	4,17
Mel+6dm+6dc	3,79	3,41	3,03	3,41	3,79
Mel+energia	3,79	3,03	3,03	3,41	3,79

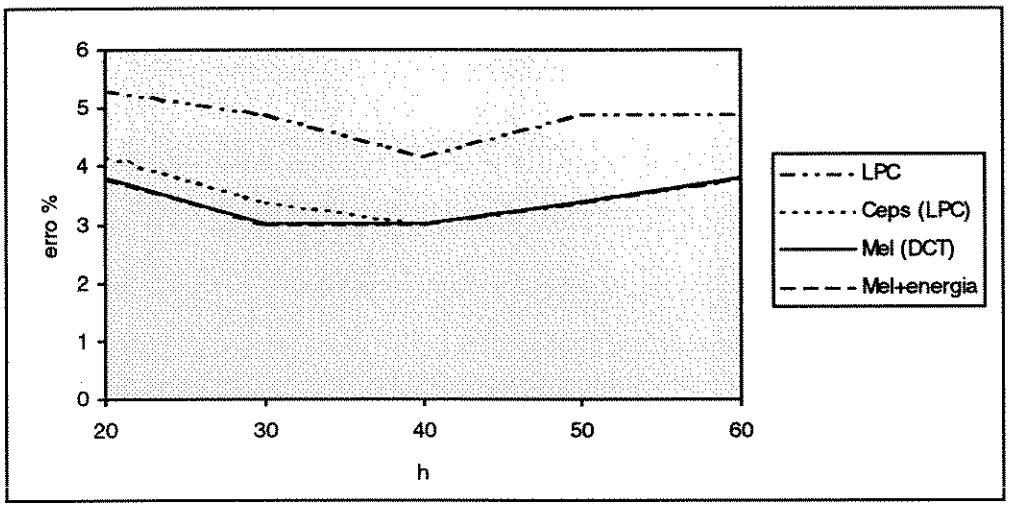


Figura 4.9: Taxas de erro em % para comandos de cálculo em função do número de neurônios da camada escondida e dos parâmetros de entrada utilizados. Análise com quadros de 30 ms recalculados a cada 20 ms (origem: Tabela 4.9).

Tabela 4.10: Taxas de erro em % para os comandos de movimento utilizando diferentes parâmetros de entrada e análise com quadros de 30 ms recalculados a cada 20 ms. Utiliza-se uma rede backpropagation com 1 camada escondida e com o número de neurônios desta camada variando de 20 a 60 (valor de h).

Parâmetros	h = 20	30	40	50	60
LPC	12,88	12,50	12,50	12,88	12,88
Ref	10,61	10,23	10,61	10,61	10,98
Ceps (LPC)	6,06	5,30	4,92	5,68	5,68
Mel (DCT)	5,30	4,92	4,17	4,92	5,30
delta-ceps (dc)	9,47	9,47	9,85	10,23	10,23
delta-mel (dm)	10,98	10,61	9,47	10,61	10,98
LPC+S	11,36	10,61	10,61	10,98	11,36
Ref+S	13,26	12,88	10,61	12,88	13,26
Ceps+S	5,30	4,92	4,17	4,92	5,30
Mel+S	5,30	4,92	4,17	4,92	5,30
Ceps+8dc	7,20	6,82	6,82	7,20	7,20
Mel+8dm	7,20	6,82	5,30	7,20	7,20
Mel+6dm+6dc	6,06	5,68	6,06	6,06	6,82
Mel+energia	4,17	3,79	3,79	4,17	4,17

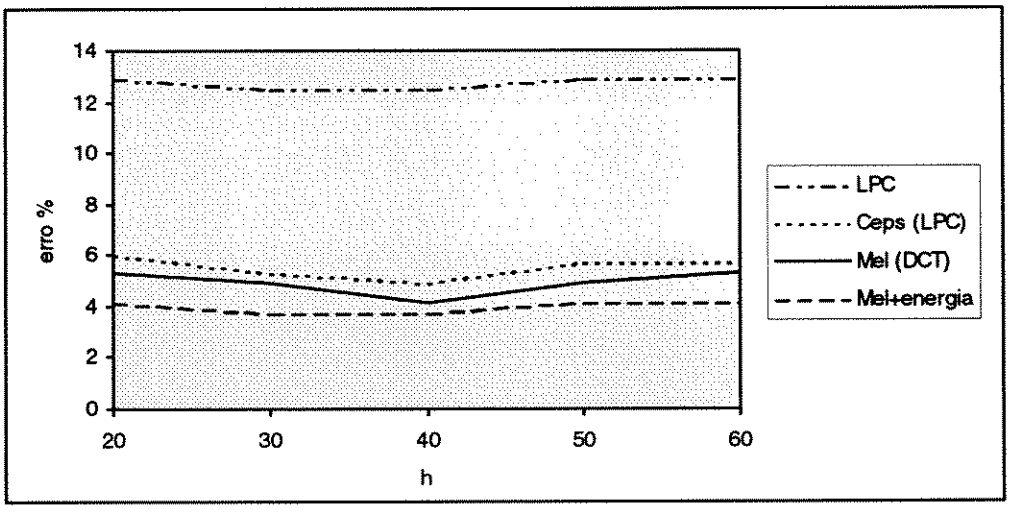


Figura 4.10: Taxas de erro em % para os comandos de movimento em função do número de neurônios da camada escondida e dos parâmetros de entrada utilizados. Análise com quadros de 30 ms recalculados a cada 20 ms (origem: Tabela 4.10).

Tabela 4.11: Taxas de erro em % para o vocabulário completo utilizando parâmetros cepstrais, mel-cepstrais e energia e análise com quadros de 30 ms recalculados a cada 20 ms. Utiliza-se uma rede backpropagation com 1 camada escondida e com o número de neurônios desta camada variando de 20 a 60 (valor de h).

Parâmetros	h = 20	30	40	50	60
Ceps (LCP)	14,06	9,77	9,51	8,98	8,59
Mel (DCT)	12,89	9,51	9,12	8,72	8,33
Mel+energia	12,37	9,25	8,85	8,33	8,07

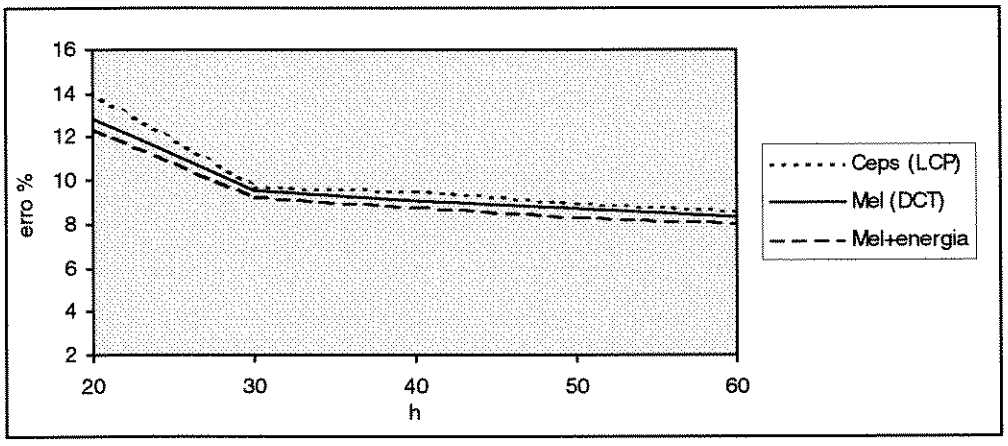


Figura 4.11: Taxas de erro em % para o vocabulário completo em função do número de neurônios da camada escondida e dos parâmetros de entrada utilizados. Análise com quadros de 30 ms recalculados a cada 20 ms (origem: Tabela 4.11).

Analisando as tabelas anteriores conclui-se que os parâmetros que produziram os melhores resultados foram, como antes, os mel-cepstrais, os cepstrais e a combinação de mel-cepstrais com energia (esta combinação foi a que produziu as menores taxas de erro). O uso de parâmetros delta ou de sonoridade não acrescentou poder discriminante à rede. A explicação para este fato é a mesma de quando se analisam todas as palavras com igual número de quadros: a inclusão de parâmetros delta não incorpora novas informações à rede devido a que esta informação já está presente na mesma como combinação linear dos parâmetros originais. De igual forma, erros na decisão sonoro/não-sonoro, assim como decisões diferentes em diversas elocuções de uma mesma palavra (por coexistirem partes sonoras e não-sonoras em um mesmo quadro), fazem com que esta informação não melhore o desempenho da rede.

Como antes, o aumento do número de neurônios da camada escondida além de 40 provoca um aumento nas taxas de erro do sistema. A explicação para este fato é a mesma que quando se analisavam todas as palavras com igual número de quadros. Quanto maior for o número de sinapses da rede, maior deve ser a seqüência de treinamento para uma estimativa acurada das sinapses. Como o número de neurônios da camada escondida foi aumentado sem aumentar-se o conjunto de treinamento, o cálculo das sinapses foi menos preciso e as taxas de erro aumentaram. Para o vocabulário completo, entretanto, como o conjunto de treinamento foi bem maior que para os vocabulários individuais, o efeito de degradação se produziu a partir de um número maior de neurônios. Assim, aumentando até 60 o número de neurônios da camada escondida as taxas de erro diminuíram.

A tabela seguinte, 4.12, é um resumo das Tabelas 4.8 a 4.11, correspondente à utilização de uma rede neural backpropagation com uma camada escondida de 30 neurônios. A figura 4.12 ilustra as taxas de erro desta rede neural em função dos parâmetros de entrada utilizados (origem: Tabela 4.12). Nesta figura, em cada grupo de colunas, a coluna mais à esquerda corresponde aos dígitos, a seguinte aos comandos de cálculo, a seguinte aos comandos de movimento e a seguinte ao vocabulário completo.

Tabela 4.12: Taxas de erro em % para os 4 vocabulários utilizando diferentes parâmetros de entrada e análise com quadros de 30 ms recalculados a cada 20 ms. Utiliza-se uma rede backpropagation com 1 camada escondida de 30 neurônios. Função de ativação utilizada: sigmóide binária.

Parâmetros	Dígitos (D)	Cálculo (C)	Movimento (M)	D+C+M
LPC	6,56	3,03	12,50	----
Ref	9,38	4,17	10,23	----
Ceps (LPC)	3,44	4,92	5,30	9,77
Mel (DCT)	2,19	3,41	4,92	9,51
Delta-ceps (dc)	8,13	4,92	9,47	----
Delta-mel (dm)	6,56	4,17	10,61	----
LPC+S	8,13	3,41	10,61	----
Ref+S	10,62	4,17	12,88	----
Ceps+S	3,44	4,92	4,92	----
Mel+S	1,88	4,17	4,92	----
Ceps+8dc	3,13	5,30	6,82	----
Mel+8dm	3,13	3,79	6,82	----
Mel+6dm+6dc	3,13	3,41	5,68	----
Mel+energia	1,88	3,03	3,79	9,25

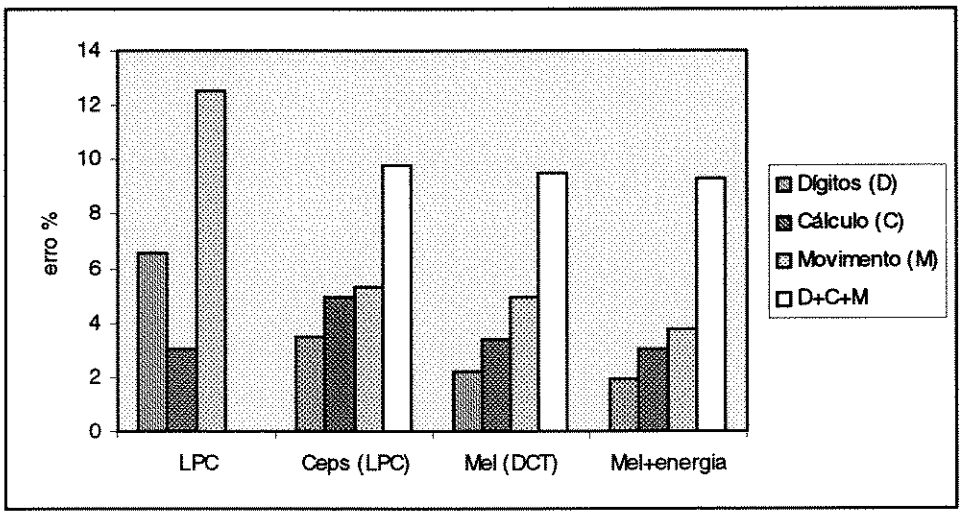


Figura 4.12: Taxas de erro em % para os 4 vocabulários em função dos parâmetros de entrada utilizados. Análise com quadros de 30 ms recalculados a cada 20 ms, e 30 neurônios na camada escondida (origem: Tabela 4.12).

Comparando o desempenho dos sistemas que utilizam igual número de quadros para todas as palavras com os que utilizam quadros de comprimento fixo, tem-se que os primeiros apresentam melhores resultados. Somente nos vocabulários de cálculo e movimento, ao se utilizar a combinação "mel-cepstrais com energia", é que a análise com quadros de comprimento fixo produziu melhores resultados. Quanto à carga computacional necessária, a análise com quadros de comprimento fixo demanda –em geral– mais cálculos, devido ao maior número de janelas de análise por palavra. Porém, sua vantagem está na possibilidade de poder realizar a extração de parâmetros à medida que o sinal de fala é obtido. Esta característica é necessária para fazer reconhecimento em tempo real. Na análise com igual número de quadros para todas as palavras, por outra parte, deve-se esperar a finalização da palavra para começar a analisar a mesma (isto é necessário já que deve-se calcular a *duração* e *posição* das diferentes janelas de análise). Se esta demora não é um fator limitador, é preferível analisar todas as palavras com igual número de quadros já que produz melhores resultados com menor esforço computacional.

4.1.3 Uso de perceptrons com duas camadas escondidas

No transcurso das simulações foi testado o uso de perceptrons multicamadas com duas camadas escondidas. Visto que a análise com igual número de quadros foi a que produziu os melhores resultados, o teste destas redes foi feito com parâmetros obtidos utilizando-se este tipo de análise. Os parâmetros empregados foram os cepstrais obtidos via LPC e os mel-cepstrais obtidos via DCT (daqui para frente só serão utilizados estes algoritmos quando se calculem coeficientes cepstrais e mel-cepstrais). O número utilizado de quadros de análise foi 40. A função de ativação empregada nos neurônios foi a sigmóide binária, colocando-se em todos os neurônios uma entrada adicional de valor 1 com sinapse treinável (bias). Como mostram as Tabelas 4.13 e 4.14 a convergência da rede foi difícil de conseguir e, quando conseguida, os resultados foram piores que para redes com uma camada escondida. A Tabela 4.13 mostra os resultados obtidos quando se utilizaram parâmetros cepstrais e a Tabela 4.14 quando se utilizaram parâmetros mel-cepstrais. **h1** corresponde ao número de neurônios da primeira camada escondida (ligada à camada de entrada), e **h2** ao da segunda camada escondida (ligada à camada de saída).

Tabela 4.13: Taxas de erro em % para os 4 vocabulários em redes backpropagation de duas camadas escondidas (h1 e h2 respectivamente). Parâmetros de entrada utilizados: cepstrais. Análise com 40 quadros por palavra. n/c: não converge.

Vocabulário	h1/h2=30/8	30/10	20/8	30/4	40/4	40/8	35/8	25/8
Dígitos (D)	7,50	8,13	n/c	n/c	9,06	10,94	10,62	13,12
Cálculo (C)	8,33	8,33	n/c	n/c	10,98	14,77	n/c	15,15
Movimento (M)	8,71	9,09	10,61	n/c	n/c	15,15	n/c	15,53
D+C+M	11,20	13,41	n/c	n/c	n/c	n/c	14,06	17,97

Tabela 4.14: Taxas de erro em % para os 4 vocabulários em redes backpropagation de duas camadas escondidas (h1 e h2 respectivamente). Parâmetros de entrada utilizados: mel-cepstrais. Análise com 40 quadros por palavra. n/c: não converge.

Vocabulário	h1/h2=30/8	30/10	20/8	30/4	40/4	40/8	35/8	25/8
Dígitos (D)	2,50	3,13	n/c	n/c	n/c	5,63	8,13	9,38
Cálculo (C)	5,30	6,44	6,44	n/c	n/c	8,33	9,09	10,23
Movimento (M)	6,06	6,44	n/c	n/c	n/c	7,96	n/c	10,61
D+C+M	10,03	10,68	n/c	n/c	n/c	13,67	n/c	16,15

Diversas outras combinações no números de neurônios de **h1** e **h2** foram testadas sem sucesso. Também foi testado o uso de 50 e 60 quadros de análise por palavra, para determinar se o problema de convergência estava associado ao número de quadros de análise utilizado. Em todos os casos os problemas persistiram. Uma explicação possível para este fato é que ao utilizar redes com duas camadas escondidas, o número de sinapses da rede aumenta sensivelmente. Sendo o número de sinapses maior, é necessário dispor, também, de um conjunto de treinamento maior para se obter uma boa estimativa das sinapses. Dado que a seqüência de treinamento utilizada com as redes de duas camadas escondidas foi a mesma que se utilizou nas redes de apenas uma camada escondida, a estimativa das sinapses não foi boa dificultando a convergência da rede. Como na época das simulações não se dispunha de uma base de dados maior, a hipótese acima não foi verificada. Assim, por causa das dificuldades para fazer convergir as redes de duas camadas escondidas e devido a que as taxas de acerto obtidas com elas foram menores que as conseguidas com as redes de uma camada escondida, no restante do trabalho empregaram-se, apenas, redes com *uma* camada escondida.

4.1.4 Teste das redes com sinais ruidosos

Até aqui os sinais utilizados para treinamento e teste das redes neurais foram os sinais originais com SNR superiores a 50 dB. Para avaliar o desempenho das redes com sinais ruidosos adicionou-

se ruído branco gaussiano aos sinais de fala de forma a atingir relações sinal/ruído de 20 dB e 10 dB por palavra (na Seção 3.1, Cap. 3, explica-se como isto foi feito). Dispôs-se, então, de três conjuntos de palavras: o de sinais originais, o de sinais com SNR = 20 dB e o de sinais com SNR = 10 dB. De posse destes sinais foram feitos três tipos de testes:

1. Treinaram-se os sistemas com sinais sem ruído e testaram-se com sinais ruidosos.
2. Treinaram-se os sistemas com sinais ruidosos e testaram-se com sinais com e sem ruído.
3. Treinaram-se os sistemas misturando sinais ruidosos e sem ruído e testaram-se com sinais ruidosos e sem ruído.

Os parâmetros utilizados nos testes foram os mel-cepstrais, os cepstrais e a combinação de mel-cepstrais com energia. Empregaram-se 40 quadros de análise por palavra. Utilizou-se uma rede backpropagation com uma camada escondida de 30 neurônios e como função de ativação dos neurônios empregou-se a sigmóide binária.

Os resultados das simulações são mostrados nas tabelas seguintes. As Tabelas 4.15 a 4.17 mostram os resultados obtidos ao se utilizar parâmetros mel-cepstrais; as Tabelas 4.18 a 4.20 ao se utilizar parâmetros cepstrais e as Tabelas 4.21 a 4.23 ao se utilizar a combinação de mel-cepstrais com energia.

Na Tabela 4.15 são mostrados os resultados obtidos ao treinar a rede neural com sinais sem ruído e testá-la com sinais ruidosos de 20 dB e 10 dB. Para efeito de comparação inclui-se também o resultado dos testes com sinais sem ruído (s/r) e a média do erro com os três tipos de sinais (20 dB, 10 dB e sem ruído).

A Tabela 4.16 mostra os resultados obtidos ao se treinar a rede com sinais ruidosos de 20 dB e testá-la com sinais de 20 dB e com sinais sem ruído. A mesma tabela mostra os resultados ao se treinar a rede com sinais ruidosos de 10 dB e testá-la com sinais de 10 dB e com sinais sem ruído.

A Tabela 4.17 mostra os resultados obtidos ao se treinar a rede misturando sinais ruidosos de 20 dB, 10 dB e sinais sem ruído, e testá-la com sinais ruidosos de 20 dB, 10 dB e com sinais sem ruído. Na tabela também são apresentadas as médias dos erros anteriores para que o leitor as compare às médias da Tabela 4.15. Os parâmetros utilizados nestas três tabelas foram os mel-cepstrais.

Tabela 4.15: Taxas de erro em % de redes backpropagation com uma camada escondida de 30 neurônios, utilizando no treinamento sinais sem ruído e no teste sinais com e sem ruído (s/r: sem ruído). Análise com 40 quadros por palavra. Coeficientes de entrada: mel-cepstrais.

Vocabulário	Conj. Teste s/r	Conj. Teste 20 dB	Conj. Teste 10 dB	Média
Dígitos (D)	1,88	20,00	42,81	21,56
Cálculo (C)	3,03	6,82	36,36	15,40
Movimento (M)	2,65	4,55	21,59	9,60
D+C+M	6,64	20,57	51,82	26,34

Tabela 4.16: Taxas de erro em % de redes backpropagation com uma camada escondida de 30 neurônios, utilizando no treinamento sinais com SNR de 20 e 10 dB, e no teste sinais com e sem ruído (s/r: sem ruído). Análise com 40 quadros por palavra. Coeficientes de entrada: mel-cepstrais.

	Conj. Treino 20 dB		Conj. Treino 10 dB	
Vocabulário	Conj. Teste s/r	Conj. Teste 20 dB	Conj. Teste s/r	Conj. Teste 10 dB
Dígitos (D)	5,31	2,50	12,81	4,69
Cálculo (C)	4,55	3,79	10,61	2,65
Movimento (M)	6,06	3,79	14,39	4,92
D+C+M	18,88	9,38	37,37	10,81

Tabela 4.17: Taxas de erro em % de redes backpropagation com uma camada escondida de 30 neurônios, utilizando no treinamento uma mistura de sinais com e sem ruído e no teste sinais com e sem ruído (s/r: sem ruído). Análise com 40 quadros por palavra. Coeficientes de entrada: mel-cepstrais.

Vocabulário	Conj. Teste s/r	Conj. Teste 20 dB	Conj. Teste 10 dB	Média
Dígitos (D)	2,19	2,81	3,75	2,92
Cálculo (C)	2,27	3,41	2,27	2,65
Movimento (M)	2,65	2,65	3,79	3,03
D+C+M	9,77	8,33	8,98	9,03

As tabelas seguintes, 4.18 a 4.20, são equivalentes às Tabelas 4.15 a 4.17, mas utilizando como entradas da rede neural coeficientes cepstrais. De igual modo as Tabelas 4.21 a 4.23 são equivalentes às Tabelas 4.15 a 4.17, utilizando como entradas da rede neural a combinação de mel-cepstrais com energia.

Tabela 4.18: Taxas de erro em % de redes backpropagation com uma camada escondida de 30 neurônios, utilizando no treinamento sinais sem ruído e no teste sinais com e sem ruído (s/r: sem ruído). Análise com 40 quadros por palavra. Coeficientes de entrada: cepstrais.

Vocabulário	Conj. Teste s/r	Conj. Teste 20 dB	Conj. Teste 10 dB	Média
Dígitos (D)	2,50	28,12	48,12	26,25
Cálculo (C)	3,41	15,91	37,12	18,81
Movimento (M)	3,41	11,74	32,20	15,78
D+C+M	8,72	32,29	62,11	34,37

Tabela 4.19: Taxas de erro em % de redes backpropagation com uma camada escondida de 30 neurônios, utilizando no treinamento sinais com SNR de 20 e 10 dB, e no teste sinais com e sem ruído (s/r: sem ruído). Análise com 40 quadros por palavra. Coeficientes de entrada: cepstrais.

	Conj. Treino 20 dB		Conj. Treino 10 dB	
Vocabulário	**Conj. Teste s/r**	**Conj. Teste 20 dB**	**Conj. Teste s/r**	**Conj. Teste 10 dB**
Dígitos (D)	8,13	4,06	21,25	4,38
Cálculo (C)	6,44	4,92	14,39	3,79
Movimento (M)	7,20	3,79	18,94	7,58
D+C+M	20,83	11,59	39,06	11,72

Tabela 4.20: Taxas de erro em % de redes backpropagation com uma camada escondida de 30 neurônios, utilizando no treinamento uma mistura de sinais com e sem ruído e no teste sinais com e sem ruído (s/r: sem ruído). Análise com 40 quadros por palavra. Coeficientes de entrada: cepstrais.

Vocabulário	**Conj. Teste s/r**	**Conj. Teste 20 dB**	**Conj. Teste 10 dB**	**Média**
Dígitos (D)	4,38	4,48	5,63	4,90
Cálculo (C)	5,68	6,44	4,55	5,56
Movimento (M)	7,58	7,58	6,82	7,32
D+C+M	12,24	10,29	10,81	11,11

Tabela 4.21: Taxas de erro em % de redes backpropagation com uma camada escondida de 30 neurônios, utilizando no treinamento sinais sem ruído e no teste sinais com e sem ruído (s/r: sem ruído). Análise com 40 quadros por palavra. Coeficientes de entrada: mel-cepstrais e energia.

Vocabulário	**Conj. Teste s/r**	**Conj. Teste 20 dB**	**Conj. Teste 10 dB**	**Média**
Dígitos (D)	2,19	21,25	43,12	22,19
Cálculo (C)	2,65	5,68	30,30	12,88
Movimento (M)	2,65	5,68	26,52	11,62
D+C+M	8,33	19,66	51,56	26,52

Tabela 4.22: Taxas de erro em % de redes backpropagation com uma camada escondida de 30 neurônios, utilizando no treinamento sinais com SNR de 20 e 10 dB, e no teste sinais com e sem ruído (s/r: sem ruído). Análise com 40 quadros por palavra. Coeficientes de entrada: mel-cepstrais e energia.

	Conj. Treino 20 dB		Conj. Treino 10 dB	
Vocabulário	**Conj. Teste s/r**	**Conj. Teste 20 dB**	**Conj. Teste s/r**	**Conj. Teste 10 dB**
Dígitos (D)	4,06	2,19	9,69	3,75
Cálculo (C)	3,41	3,41	5,68	3,66
Movimento (M)	6,44	4,92	15,53	4,92
D+C+M	16,54	7,16	32,55	10,16

Tabela 4.23: Taxas de erro em % de redes backpropagation com uma camada escondida de 30 neurônios, utilizando no treinamento uma mistura de sinais com e sem ruído e no teste sinais com e sem ruído (s/r: sem ruído). Análise com 40 quadros por palavra. Coeficientes de entrada: mel-cepstrais e energia.

Vocabulário	**Conj. Teste s/r**	**Conj. Teste 20 dB**	**Conj. Teste 10 dB**	**Média**
Dígitos (D)	3,44	2,81	3,75	3,33
Cálculo (C)	3,79	5,68	1,89	3,79
Movimento (M)	5,68	5,68	4,17	5,18
D+C+M	7,55	6,38	7,81	7,25

Pode-se verificar nas Tabelas 4.15 a 4.23 que o desempenho das redes backpropagation degrada-se na presença de ruído, como é previsível, mas que esta degradação é pequena se os conjuntos de treinamento e teste possuem relações sinal/ruído semelhantes. Assim, nas Tabelas 4.15, 4.18 e 4.21 vemos que redes treinadas com sinais de SNR = 20 dB possuem erros na faixa de 2,19% a 11,59% ao serem testadas com sinais de SNR = 20 dB, mas os erros sobem até 20,83% quando são testadas com sinais de SNR diferente. De igual forma nas Tabelas 4.16, 4.19 e 4.22 vemos que as redes treinadas com sinais de SNR = 10 dB possuem erros na faixa de 2,65% a 11,72% quando são testadas com sinais de SNR = 10 dB, mas os erros sobem a valores entre 5,68% e 39,06% quando são testadas com sinais de SNR diferente. Isto implica em duplicar o erro no melhor dos casos e praticamente quadruplicá-lo no pior dos casos.

Para redes treinadas com sinais sem ruído e testadas com sinais também sem ruído, as taxas de erro variaram de 1,88% a 8,72%, enquanto que testadas com sinais ruidosos os erros variaram de 4,55% a 62,11%.

Finalmente, as redes treinadas com uma mistura de sinais ruidosos e sem ruído tiveram um comportamento bastante uniforme e bem melhor que o das anteriores. As Tabelas 4.17, 4.20 e 4.23 mostram que os erros de teste nestas redes com sinais sem ruído variaram entre 2,19% e 12,24% e para sinais ruidosos variaram entre 1,89% e 10,81%. Portanto, os erros ficaram bem próximos entre si. Adicionalmente, comparando-se as médias das Tabelas 4.15 e 4.17, as médias das Tabelas 4.18 e 4.20 e as médias das Tabelas 4.21 e 4.23, verifica-se que o fato de treinar as redes com uma mistura de sinais ruidosos e sem ruído, faz com que os erros de reconhecimento sejam bem menores que quando são treinadas com um tipo de sinal e testadas ou utilizadas com outro.

Desta forma podemos concluir que para melhorar o comportamento da rede em ambientes ruidosos, o conjunto de treinamento deve incluir sinais de fala obtidos em um ambiente tão próximo quanto possível daquele no qual a rede será utilizada.

4.2 Uso de redes de Kohonen, Kohonen-Grossberg e LVQ

Nesta seção apresentar-se-ão os resultados obtidos durante as simulações de sistemas de reconhecimento de palavras isoladas, utilizando redes neurais de Kohonen, Kohonen-Grossberg e LVQ.

Visto que a análise com igual número de quadros para todas as palavras resultou em taxas de reconhecimento maiores que as obtidas com quadros de comprimento fixo (Seções 4.1.1 e 4.1.2), as redes neurais desta seção foram testadas utilizando-se análise com igual número de quadros em todas as palavras. O número de quadros utilizado foi 40 e os parâmetros empregados como entrada das redes neurais foram os cepstrais, os mel-cepstrais e a combinação de mel-cepstrais com energia.

No Cap. 2, Seções 2.4.3, 2.4.4 e 2.4.5 foram estudadas as redes de Kohonen, Kohonen-Grossberg e LVQ respectivamente. Lembrando brevemente, nas redes de Kohonen o treinamento era *não-supervisionado*. Os mapas eram usualmente lineares ou quadrados e as posições iniciais dos vetores eram aleatórias ou arbitrárias[1]. O algoritmo de treinamento denominava-se SOM (*self-organizaing map*).

Nas redes LVQ o treinamento era *supervisionado*. A posição inicial dos vetores podia ser estabelecida tomando-se os primeiros m vetores do conjunto de treinamento (cada um representando uma classificação conhecida e diferente), ou colocando os vetores em posições aleatórias e utilizando o algoritmo SOM para definir suas posições iniciais. Após a fase SOM aplicava-se algum dos algoritmos LVQ (LVQ1, LVQ2, LVQ2.1 ou LVQ3).

Quanto às redes de Kohonen-Grossberg, estas eram redes onde o treinamento era em parte *não-supervisionado* e em parte *supervisionado*. Durante a fase não-supervisionada do treinamento aplicava-se o algoritmo SOM na camada de Kohonen. Durante a fase supervisionada mapeavam-se as saídas da camada de Kohonen nas saídas desejadas na camada de Grossberg.

As redes aqui simuladas foram as seguintes:

1. Redes de Kohonen, onde aos vetores resultantes aplicou-se o algoritmo LVQ para otimizar suas posições (vetores de referência).
2. Redes LVQ com posição inicial dos vetores dada: a) pelos primeiros exemplos do conjunto de treinamento, b) pela aplicação do algoritmo SOM aos vetores em posições aleatórias.
3. Redes de Kohonen-Grossberg, aplicando-se o algoritmo LVQ na camada de Kohonen como no caso 1.

[1] Ao falarmos aqui de *vetor*, estamos nos referindo ao *vetor que representa a palavra*, e não ao vetor resultante de um quadro de análise do sinal. Deve ser lembrado que neste trabalho estamos fazendo reconhecimento estático da palavra falada. Portanto, os vetores resultantes da análise da palavra são concatenados num único vetor que representa a palavra e que alimenta a rede neural.

Deve-se notar que as redes do caso 2.b são equivalentes às do caso 1. Assim, os resultados das redes LVQ apresentados na próxima seção (4.2.1), correspondem também aos das redes de Kohonen. Na Seção 4.2.2 apresentar-se-ão os resultados das redes de Kohonen-Grossberg.

4.2.1 Redes de Kohonen e LVQ

Como foi mencionado, as redes LVQ podem ter seus vetores inicializados pelos primeiros m vetores do conjunto de treinamento (cada um representando uma classificação conhecida e diferente), ou colocando-se os vetores em posições aleatórias e utilizando o algoritmo SOM para definir suas posições iniciais. Posteriormente aplica-se algum dos algoritmos LVQ (LVQ1, LVQ2, LVQ2.1 ou LVQ3). Para a fase SOM deve definir-se o tipo de mapa a utilizar (linear ou quadrado) e o número de neurônios do mesmo.

Dos quatro algoritmos LVQ estudados na Seção 2.4.5 (LVQ1, LVQ2, LVQ2.1 e LVQ3), o implementado aqui foi o LVQ3 por ser a evolução dos anteriores. A janela que utiliza este algoritmo foi definida com $\varepsilon = 0{,}3$ (foram feitos testes com $\varepsilon = 0{,}2$ e $\varepsilon = 0{,}4$, mas os melhores resultados se obtiveram com $\varepsilon = 0{,}3$). A taxa de aprendizagem $\eta(t)$ utilizada quando os vetores de referência pertencem a classes diferentes iniciou-se com valor 0,01, sendo diminuída época a época com a lei $\eta(t+1) = 0{,}6.\eta(t)$. A taxa de aprendizagem $\beta(t) = q.\eta(t)$, utilizada quando os vetores pertencem à mesma classe, foi calculada usando $q = 0{,}1$. Em ambos os casos t refere-se à **época** de treinamento. Para os casos em que se utilizou o algoritmo SOM, na fase de ordenamento do mapa empregaram-se 4 épocas para os dígitos, comandos de cálculo e de movimento; e 2 para o vocabulário completo. Na fase de convergência do mapa empregaram-se 10 épocas nos 4 vocabulários.

Quanto à lei de variação da taxa de aprendizagem, durante a fase de ordenamento do mapa utilizou-se $\eta(t) = 0{,}9\ (1 - t/2000)$ onde t indica a iteração (passos 2 e 3 do algoritmo de treinamento; Seção 2.4.3). Durante a fase de convergência utilizou-se $\eta(t) = 0{,}01$. Para o raio de vizinhança, durante a fase de ordenamento começou-se incluindo todos os neurônios da rede, diminuindo $r(t)$ gradualmente até 1 com a equação $r(t) = r(0)\ (1 - t/2000)$. Durante a fase de convergência utilizou-se $r(t) = 1$ ou $r(t) = 0$.

O número de neurônios testado nas redes Kohonen/LVQ foi o seguinte:

- ***para os dígitos:*** no caso de inicializar os vetores com exemplos de treinamento se utilizaram 10 neurônios; no caso de usar o algoritmo SOM (Kohonen), utilizaram-se 10 e 20 neurônios em mapas lineares e 16 e 25 neurônios em mapas quadrados;
- ***para comandos de cálculo e movimento:*** no caso de inicializar os vetores com exemplos de treinamento se utilizaram 11 neurônios; no caso de usar o algoritmo SOM utilizaram-se 11 e 22 neurônios em mapas lineares e 16 e 25 neurônios em mapas quadrados;
- ***para o vocabulário completo:*** no caso de inicializar os vetores com exemplos de treinamento se utilizaram 32 neurônios; no caso de usar o algoritmo SOM utilizaram-se 32 neurônios em mapas lineares e 36 e 49 neurônios em mapas quadrados.

Escolhendo-se tantos neurônios como classes (neste caso palavras a reconhecer), teremos um neurônio representando cada classe. Escolhendo-se mais neurônios do que classes, haverá classes representadas por mais que um neurônio. O número de entradas da rede foi 480 quando se utilizaram parâmetros cepstrais ou mel-cepstrais e 520 quando se utilizou a combinação mel-cepstrais com energia. As tabelas e figuras seguintes mostram os resultados obtidos.

Tabela 4.24: Taxas de erro em % para os dígitos utilizando redes LVQ. Para a fase SOM os mapas são lineares de 10 e 20 neurônios e quadrados de 16 e 25 neurônios. Parâmetros utilizados: cepstrais, mel-cepstrais e a combinação "mel-cepstrais com energia". Análise com 40 quadros por palavra.

Mapa	Cepstrais	Mel	Mel + energia
Sem mapa, 10 neurônios	3,13	3,13	3,13
Linear, 10 neurônios	3,13	3,13	2,81
Linear, 20 neurônios	2,81	3,44	3,44
Quadrado, 16 neurônios	2,81	2,81	3,13
Quadrado, 25 neurônios	3,75	3,13	3,13

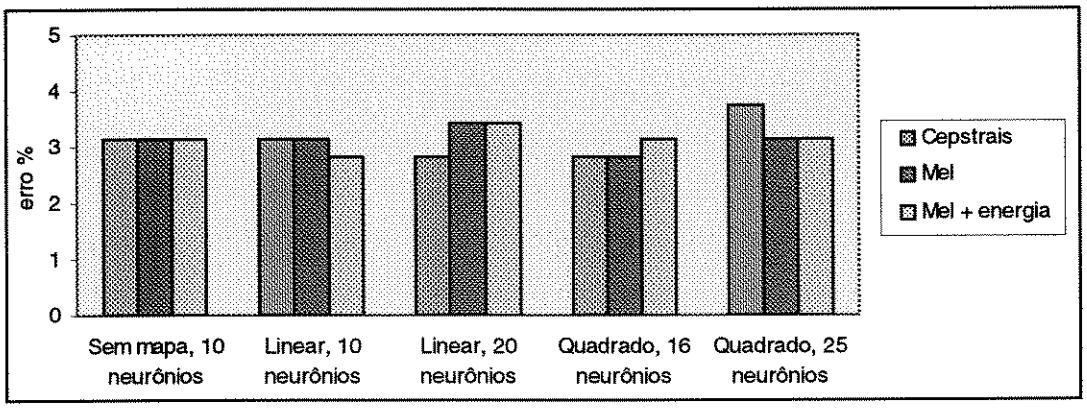


Figura 4.13: Taxas de erro em % para os dígitos utilizando redes LVQ. Origem: Tabela 4.24.

Tabela 4.25: Taxas de erro em % para os comandos de cálculo utilizando redes LVQ. Para a fase SOM os mapas são lineares de 11 e 22 neurônios e quadrados de 16 e 25 neurônios. Parâmetros utilizados: cepstrais, mel-cepstrais e a combinação "mel-cepstrais com energia". Análise com 40 quadros por palavra.

Mapa	Cepstrais	Mel	Mel + energia
Sem mapa, 11 neurônios	4,17	3,79	3,03
Linear, 11 neurônios	4,17	3,41	2,65
Linear, 22 neurônios	4,55	4,17	3,03
Quadrado, 16 neurônios	4,17	3,79	4,17
Quadrado, 25 neurônios	4,55	3,41	4,17

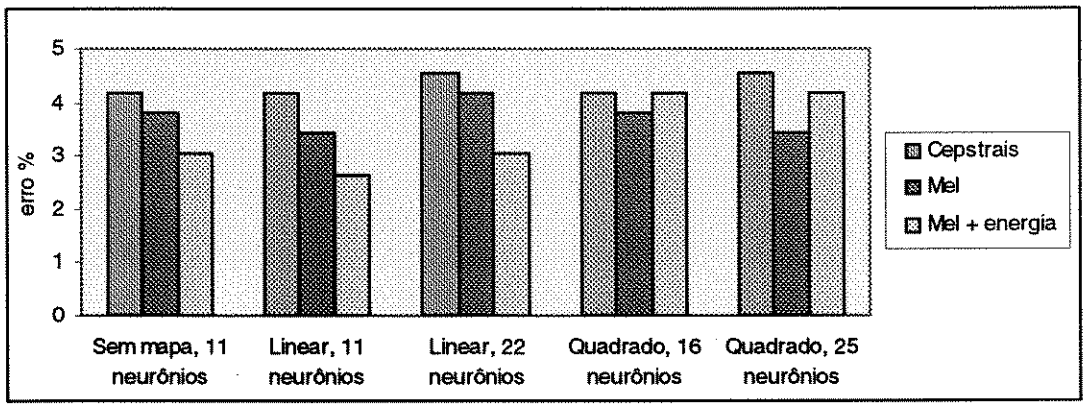


Figura 4.14: Taxas de erro em % para os comandos de cálculo utilizando redes LVQ. Origem: Tabela 4.25.

Tabela 4.26: Taxas de erro em % para os comandos de movimento utilizando redes LVQ. Para a fase SOM os mapas são lineares de 11 e 22 neurônios e quadrados de 16 e 25 neurônios. Parâmetros utilizados: cepstrais, mel-cepstrais e a combinação "mel-cepstrais com energia". Análise com 40 quadros por palavra.

Mapa	Cepstrais	Mel	Mel + energia
Sem mapa, 11 neurônios	4,17	3,79	3,41
Linear, 11 neurônios	4,55	2,65	3,03
Linear, 22 neurônios	4,17	3,41	2,65
Quadrado, 16 neurônios	4,17	3,03	3,41
Quadrado, 25 neurônios	3,79	3,79	3,41

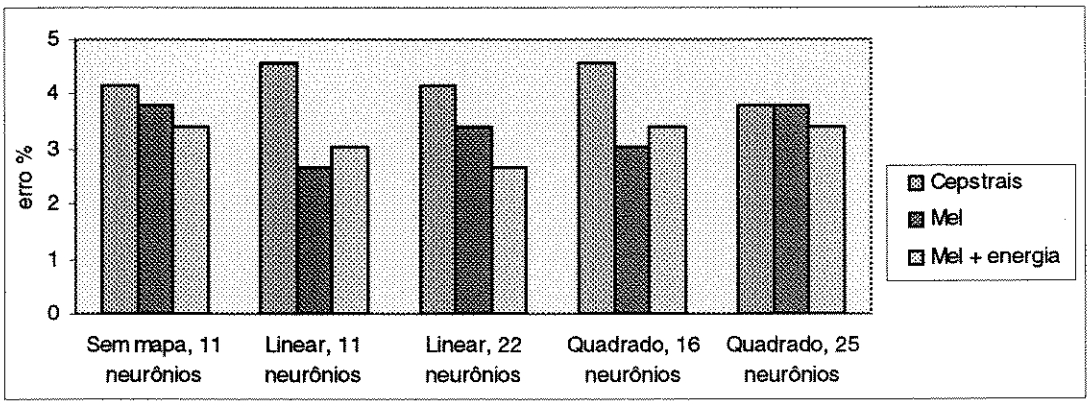


Figura 4.15: Taxas de erro em % para os comandos de movimento utilizando redes LVQ. Origem: Tabela 4.26.

Tabela 4.27: Taxas de erro em % para o vocabulário completo utilizando redes LVQ. Para a fase SOM os mapas são lineares de 32 neurônios e quadrados de 36 e 49 neurônios. Parâmetros utilizados: cepstrais, mel-cepstrais e a combinação "mel-cepstrais com energia". Análise com 40 quadros por palavra.

Mapa	Cepstrais	Mel	Mel + energia
Sem mapa, 32 neurônios	10,51	8,50	8,50
Linear, 32 neurônios	10,38	8,63	8,50
Quadrado, 36 neurônios	10,51	8,72	8,63
Quadrado, 49 neurônios	10,72	8,63	8,63

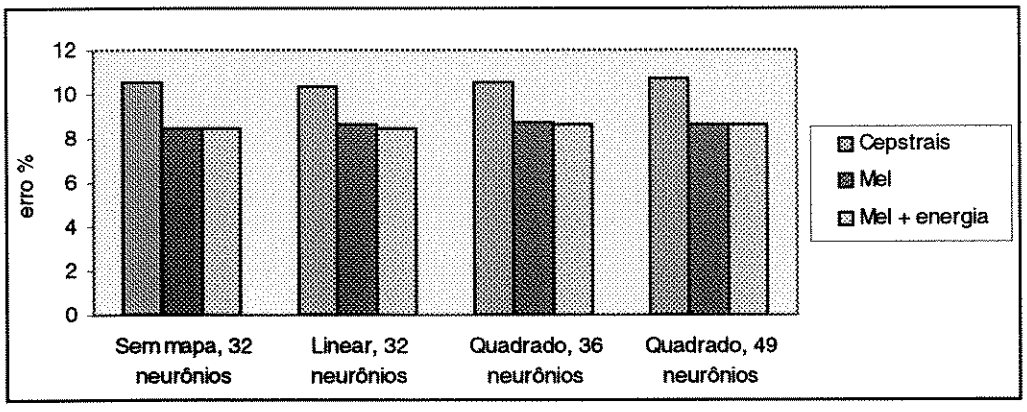


Figura 4.16: Taxas de erro em % para o vocabulário completo utilizando redes LVQ. Origem: Tabela 4.27.

Das tabelas e figuras acima pode ser visto que praticamente não há diferença entre usar parâmetros mel-cepstrais e a combinação de mel-cepstrais com energia. O desempenho dos sistemas com estes parâmetros é muito parecido. Os cepstrais, entretanto, tem um comportamento inferior. Quanto ao tipo de configuração/mapa a usar, também aqui não há diferenças substanciais. Para os dígitos as melhores configurações foram as que usaram mais neurônios na camada de Kohonen, do que palavras a reconhecer. Para os comandos de cálculo e de movimento e para o vocabulário completo, os resultados com uma ou outra configuração foram praticamente iguais. O fato do comportamento ser similar, pode ser justificado pelo fato de que os conjuntos de treinamento são muito pequenos e, em conseqüência, os mapas formados não são precisos. Desta forma, colocar mais neurônios para representar uma mesma classe não introduz melhorias.

Se comparamos o desempenho das redes de Kohonen com o dos perceptrons multicamada, verifica-se que as redes de Kohonen tiveram um desempenho inferior. As taxas de erro obtidas com estas redes foram maiores que com os perceptrons multicamada e a sua convergência foi mais demorada (os tempos de treinamento foram até 10 vezes maiores). Em relação aos tempos de reconhecimento, as redes de Kohonen também demoraram entre duas e três vezes mais.

Nas simulações seguintes as redes de Kohonen foram testadas com sinais ruidosos (sinais com SNR de 20 dB e 10 dB). Todas as configurações anteriores foram avaliadas. Na maior parte dos casos, os melhores resultados se obtiveram com as redes que utilizaram igual número de neurônios do que classes. Os resultados das Tabelas 4.28 a 4.30 correspondem a estas configurações. Inclui-se nas tabelas, como referência, as taxas de erro obtidas com sinais sem ruído e o valor médio dos três erros. A Tabela 4.28 corresponde ao uso de parâmetros cepstrais, a 4.29 ao uso de mel-cepstrais e a 4.30 ao uso de mel-cepstrais com energia.

Tabela 4.28: Taxas de erro em % para os 4 vocabulários utilizando redes LVQ com sinais com e sem ruído (s/r). Parâmetros utilizados: cepstrais. Conjunto de treinamento sem ruído. Análise com 40 quadros por palavra.

Vocabulário / Neurônios	Conj. Teste s/r	Conj. Teste 20 dB	Conj. Teste 10 dB	Média
Dígitos / 10	3,13	21,25	43,12	22,50
Cálculo / 11	4,17	23,48	50,38	26,01
Movimento / 11	4,17	22,73	61,36	29,42
D+C+M / 32	10,38	35,42	74,44	40,08

Tabela 4.29: Taxas de erro em % para os 4 vocabulários utilizando redes LVQ com sinais com e sem ruído (s/r). Parâmetros utilizados: mel-cepstrais. Conjunto de treinamento sem ruído. Análise com 40 quadros por palavra.

Vocabulário / Neurônios	Conj. Teste s/r	Conj. Teste 20 dB	Conj. Teste 10 dB	Média
Dígitos / 10	3,13	18,75	41,56	21,15
Cálculo / 11	3,41	17,80	54,92	25,38
Movimento / 11	2,65	12,25	34,47	16,46
D+C+M / 32	8,50	27,34	62,63	32,82

Tabela 4.30: Taxas de erro em % para os 4 vocabulários utilizando redes LVQ com sinais com e sem ruído (s/r). Parâmetros utilizados: mel-cepstrais e energia. Conjunto de treinamento sem ruído. Análise com 40 quadros por palavra.

Vocabulário / Neurônios	Conj. Teste s/r	Conj. Teste 20 dB	Conj. Teste 10 dB	Média
Dígitos / 10	2,81	19,69	43,44	21,98
Cálculo / 11	2,65	19,32	54,92	25,63
Movimento / 11	3,03	14,64	36,36	18,01
D+C+M / 32	8,50	28,52	70,14	35,72

Pode ser visto nas tabelas anteriores que as taxas de erro aumentam drasticamente quando as redes de Kohonen/LVQ são treinadas com sinais sem ruído e testadas com sinais ruidosos. O aumento nas taxas de erro é maior que o ocorrido nas redes backpropagation (Seção 4.1.4, Tabelas 4.15, 4.18 e 4.21). Isto indica maior sensibilidade ao ruído nas redes de Kohonen/LVQ que nas redes backpropagation.

No caso de treinar as redes com uma mistura de sinais sem ruído e sinais ruidosos (SNR de 20 dB e 10 dB), o comportamento das redes melhora sensivelmente. As Tabelas 4.31 a 4.33 mostram que, neste caso, as taxas de erro têm valores similares para todos os sinais e que estes valores são menores que os correspondentes das tabelas anteriores. Tal comportamento é similar ao ocorrido nas redes backpropagation.

Tabela 4.31: Taxas de erro em % para os 4 vocabulários utilizando redes LVQ com sinais com e sem ruído (s/r). Parâmetros utilizados: cepstrais. Conjunto de treinamento: mistura de sinais com e sem ruído. Análise com 40 quadros por palavra.

Vocabulário / Neurônios	Conj. Teste s/r	Conj. Teste 20 dB	Conj. Teste 10 dB	Média
Dígitos / 10	3,44	4,06	5,00	4,17
Cálculo / 11	6,44	8,33	5,30	6,69
Movimento / 11	7,20	6,06	6,06	6,44
D+C+M / 32	14,58	13,93	14,45	14,32

Tabela 4.32: Taxas de erro em % para os 4 vocabulários utilizando redes LVQ com sinais com e sem ruído (s/r). Parâmetros utilizados: mel-cepstrais. Conjunto de treinamento: mistura de sinais com e sem ruído. Análise com 40 quadros por palavra.

Vocabulário / Neurônios	Conj. Teste s/r	Conj. Teste 20 dB	Conj. Teste 10 dB	Média
Dígitos / 10	3,44	3,85	4,59	3,96
Cálculo / 11	4,17	4,55	4,55	4,42
Movimento / 11	4,17	4,42	4,67	4,42
D+C+M / 32	11,33	10,55	11,20	11,03

Tabela 4.33: Taxas de erro em % para os 4 vocabulários utilizando redes LVQ com sinais com e sem ruído (s/r). Parâmetros utilizados: mel-cepstrais e energia. Conjunto de treinamento: mistura de sinais com e sem ruído. Análise com 40 quadros por palavra.

Vocabulário / Neurônios	Conj. Teste s/r	Conj. Teste 20 dB	Conj. Teste 10 dB	Média
Dígitos / 10	3,44	4,06	4,69	4,06
Cálculo / 11	4,17	4,17	3,03	3,79
Movimento / 11	6,44	3,03	3,41	4,29
D+C+M / 32	11,46	10,55	11,33	11,11

4.2.2 Redes de Kohonen-Grossberg

Nesta seção apresentar-se-ão os resultados obtidos com as redes de Kohonen-Grossberg. As redes simuladas são as da seção anterior que utilizaram um número de neurônios igual ao número de classes a reconhecer, às quais se acrescentou a camada de Grossberg.

Para treinar a camada de Grossberg foi utilizada a Eq. 2.17 do Cap. 2. O valor inicial da taxa de aprendizagem foi fixado em 0,1. Durante o aprendizado corrigiu-se este valor época a época com a equação $\beta(época + 1) = 0{,}6\ \beta(época)$.

As tabelas seguintes mostram os resultados obtidos. A Tabela 4.34 corresponde ao uso de sinais sem ruído no treinamento e testes, e a 4.35 a uma mistura de sinais ruidosos e sem ruído tanto no treinamento como nos testes (para sinais ruidosos SNR = 10 dB e 20 dB). Os parâmetros de entrada utilizados foram os cepstrais, os mel-cepstrais e a combinação de mel-cepstrais com energia.

Tabela 4.34: Taxas de erro em % para os diferentes vocabulários utilizando redes de Kohonen-Grossberg. Parâmetros utilizados: cepstrais, mel-cepstrais e a combinação "mel-cepstrais com energia". Análise com 40 quadros por palavra. Treinamento e teste com sinais sem ruído.

Vocabulário / Neurônios	Cepstrais	Mel	Mel + energia
Dígitos / 10	3,23	3,13	3,23
Cálculo / 11	4,55	4,17	3,03
Movimento / 11	4,55	2,78	3,28
D+C+M / 32	10,51	8,55	8,50

Tabela 4.35: Taxas de erro em % para os diferentes vocabulários utilizando redes de Kohonen-Grossberg. Parâmetros utilizados: cepstrais, mel-cepstrais e a combinação "mel-cepstrais com energia". Análise com 40 quadros por palavra. Treinamento e teste com uma mistura de sinais ruidosos e sem ruído.

Vocabulário / Parâmetros	Conj. Teste s/r	Conj. Teste 20 dB	Conj. Teste 10 dB	Média
Dígitos / Ceps	3,54	4,06	5,10	4,23
Dígitos / Mel	3,34	3,85	4,48	3,89
Dígitos / Mel + E	3,75	3,85	4,69	4,10
Cálculo / Ceps	6,82	8,71	6,44	7,32
Cálculo / Mel	4,30	4,80	4,92	4,67
Cálculo / Mel + E	4,30	4,55	4,67	4,51
Movimento / Ceps	7,33	6,19	6,31	6,61
Movimento / Mel	4,55	4,67	5,30	4,84
Movimento / Mel + E	6,57	3,16	3,28	4,34
D+C+M / Ceps	15,10	14,28	14,97	14,78
D+C+M / Mel	11,98	11,07	11,59	11,55
D+C+M / Mel + E	12,02	10,85	11,89	11,59

Comparando as Tabelas 4.28 a 4.33 com as Tabelas 4.34 e 4.35 verifica-se que ao acrescentar a camada de Grossberg, as taxas de erro se incrementam levemente. Isto é devido aos erros que esta camada introduz degradando o desempenho original da rede.

Visto que com as redes de Kohonen e de Kohonen-Grossberg obtiveram-se taxas de erro maiores que as obtidas com as redes backpropagation, e que, adicionalmente, o treinamento delas foi bem mais demorado, o uso de redes de Kohonen ou Kohonen-Grossberg não é aconselhado para nossa aplicação.

4.3 Uso de Quantização Vetorial

Até aqui as redes neurais simuladas foram alimentadas com os vetores de parâmetros originais obtidos da análise do sinal de fala. A idéia a testar nesta seção é a de fazer uma quantização vetorial destes vetores e alimentar a rede neural com os vetores quantizados ao invés de com os originais. Isto incrementa o poder de generalização do sistema de reconhecimento, já que as diferentes realizações de um mesmo som terão os mesmos vetores de quantização, ou vetores muito próximos[1]. Deve-se verificar, porém, se as taxas de reconhecimento melhoram.

"Quantizar é aproximar sinais de amplitudes contínuas por sinais de amplitudes discretas. Um quantizador vetorial K-dimensional de N níveis é um processo que determina para cada vetor $\mathbf{x} = (x_1,\ldots, x_K)$, um vetor $\mathbf{x}_i = q(\mathbf{x})$, o qual pertence a um alfabeto $\mathbf{A} = \{\mathbf{x}_i\}$, $i = 1,\ldots, N$. O alfabeto **A** é denominado *codebook*, N é o número de vetores código e cada $\mathbf{x}_i = (x_{i1},\ldots, x_{iK})$ é um vetor código" (Martins, 1997).

No processo de quantização, o vetor a quantizar, **x**, é comparado com cada um dos vetores código empregando-se alguma medida de distorção. O vetor código que resulta na menor distorção é escolhido para representar o vetor **x**. Neste trabalho, utilizou-se como medida de distorção a distância euclidiana.

Usualmente um quantizador vetorial (QV) devolve os índices de vetores e não os vetores em si. A través do codebook (CB), é possível mapear os índices dos vetores nos vetores correspondentes. Isto é o que foi feito aqui para poder alimentar a rede neural com os vetores quantizados e não com os índices.

O algoritmo utilizado para a criação dos CB foi o LBG (Linde-Buzo-Gray) com busca exaustiva (Linde, et al., 1980). Este algoritmo utiliza o procedimento *"K-means"* e a técnica *"splitting"*. O algoritmo divide a seqüência de treinamento em N células e satisfaz as condições necessárias para ser considerado ótimo. Como medida de distorção utilizou-se a distância euclidiana. Quando a diferença entre as distorções médias de duas iterações consecutivas era menor que 1%, suspendia-se a execução do algoritmo considerando-se criado o CB.

[1] Assume-se aqui que diferentes realizações de um mesmo som geram vetores de parâmetros suficientemente parecidos, de forma que o quantizador vetorial lhes atribui o mesmo vetor código ou vetores código próximos.

Foram criados CB de 16, 32, 64, 128 e 256 vetores. Cada vocabulário teve sua própria seqüência de treinamento para criação dos CB. O número de vetores (seqüência de treinamento) utilizado para criação dos CB foi o seguinte: 38400 vetores para os dígitos (4 elocuções/palavra), 31680 vetores para os comandos de cálculo, 31680 para os de movimento (3 elocuções/palavra), e 92160 vetores para o vocabulário completo (3 elocuções/palavra).

Os vetores de treinamento foram obtidos analisando-se todas as palavras com igual número de quadros (quarenta), e utilizando 1/3 de superposição entre quadros consecutivos. Os quadros foram ponderados com janelas de Hamming.

A quantização vetorial foi testada em redes backpropagation com uma camada escondida contendo 30 e 50 neurônios. Utilizaram-se sinais sem ruído e parâmetros mel-cepstrais (12 por quadro ou vetor).

Os resultados obtidos são mostrados na tabela seguinte. Nesta tabela, **CB** indica o tamanho do codebook (número de vetores código), e **h** o número de neurônios da camada escondida.

Tabela 4.36: Taxas de erro em % para os 4 vocabulários utilizando quantização vetorial e redes backpropagation com uma camada escondida. Análise com 40 quadros por palavra. Codebooks de 16, 32, 64, 128 e 256 vetores. h (neurônios da camada escondida): 30 e 50.

	Dígitos (D)		Cálculo (C)		Movimento (M)		D+C+M	
CB	h = 30	50	h = 30	50	h = 30	50	h = 30	50
16	2,50	2,81	3,79	4,17	3,79	3,79	12,24	12,37
32	2,81	3,13	3,03	2,65	5,68	6,06	13,41	13,80
64	3,13	3,44	2,27	2,27	3,41	3,79	9,12	10,16
128	3,13	3,13	2,27	2,65	3,03	3,41	7,16	7,29
256	2,81	2,81	3,03	3,41	3,41	3,41	10,29	10,29

Os resultados que se obtiveram com 30 neurônios escondidos foram levemente superiores aos obtidos com 50 neurônios. Novamente isto pode ser explicado pelo tamanho limitado da seqüência de treinamento disponível. Quanto à técnica de quantização vetorial em si, os resultados não superaram os obtidos na Seção 4.1.1 quando se realizou o mesmo tipo de análise (igual número de quadros em todas as palavras), mas sem quantização vetorial. Assim, não é interessante sua utilização para nossa aplicação.

4.4 Uso de *Trace Segmentation* e *Individual Trace Segmentation*

Trace Segmentation (TS) e *Individual Trace Segmentation* (ITS) (Cabral e Tattersall, 1995), são técnicas que permitem realizar uma normalização temporal não-linear do sinal de fala. Isto facilita a comparação entre padrões de durações diferentes. A normalização permite utilizar quadros de comprimento fixo na análise do sinal de voz mantendo –porém– constante, o número de entradas da rede neural.

Nesta seção apresentar-se-ão os resultados obtidos ao utilizar-se as técnicas TS e ITS em redes backpropagation e LVQ. A análise foi feita com quadros de 30 ms recalculando-se os parâmetros a cada 4ms e a cada 2 ms. Os quadros foram ponderados com janelas de Hamming. Os sinais foram pré-enfatizados com um fator de 0,9. O desempenho dos sistemas foi medido com sinais sem ruído.

4.4.1 Técnicas TS e ITS

Quando se analisam as palavras com quadros de comprimento fixo, uma das opções possíveis para manter constante o número de entradas da rede neural é dizimar/interpolar o número de quadros obtido, de forma a atingir os n quadros que a rede aceita. *Trace Segmentation* e *Individual Trace Segmentation* são técnicas que *dizimam* o número de quadros de uma palavra e exigem, portanto, que na análise da palavra se obtenha um número de quadros maior que os desejados.

Lembrando que cada quadro de análise gera um vetor de p elementos, a seqüência de vetores que representa uma palavra descreve uma trajetória no hiperespaço de dimensão p. A idéia por trás de TS é a seguinte: dado que cada classe ou palavra do vocabulário a reconhecer tem uma trajetória espacial diferente, pode-se codificar a forma desta trajetória escolhendo-se n pontos igualmente espaçados na curva. Cada trajetória é, então, representada com n pontos, e os vetores correspondentes aos n pontos são os que alimentarão a rede neural. O processo de escolher n pontos igualmente espaçados sobre a curva ou trajetória é o que chamamos de *Trace Segmentation* (a trajetória é dividida em segmentos). A técnica permite absorver variações temporais entre diferentes elocuções de uma mesma palavra, já que se produz uma normaliza-

ção não-linear da trajetória no tempo (todas as trajetórias passam a ser representadas com o mesmo número de pontos). A exatidão da trajetória original dependerá de quão freqüentemente o sinal de fala seja analisado. Quanto menor for o intervalo de tempo *ts* entre cálculo de parâmetros, mais precisa será a trajetória.

As figuras 4.17 e 4.18 ajudam a entender estes conceitos. Na Fig. 4.17 é mostrada a trajetória correspondente a uma palavra supondo que os vetores de análise **v** têm apenas duas componentes: a_1 e a_2 ($p = 2$). A trajetória obtém-se calculando o vetor $\mathbf{v} = (a_1, a_2)$ a cada *ts* ms.

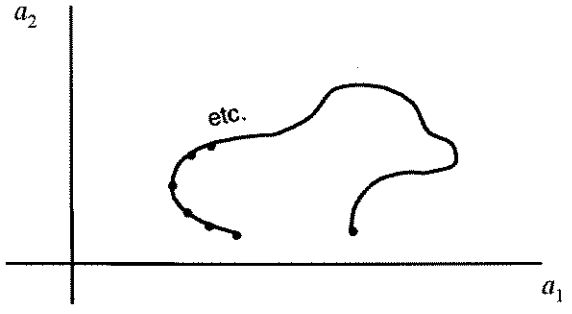


Figura 4.17: Trajetória correspondente a uma palavra, descrita por uma seqüência de vetores de dimensão 2.

Supondo a existência de 9 quadros de análise, a Fig. 4.18 mostra a trajetória da Fig. 4.17 *esticada* (linha superior), e os pontos $\mathbf{v}_0$ a $\mathbf{v}_8$ obtidos ao analisar a palavra. Em cima dos pontos $\mathbf{v}_0$ a $\mathbf{v}_8$ indicam-se os tempos t_0 a t_8 nos quais foram calculados os vetores anteriores. Dado que a análise é realizada a cada *ts* ms, "$t_{i+1} - t_i = ts$" para todo i. Indicam-se também na trajetória superior, as distâncias l_i entre os pontos $\mathbf{v}_i$.

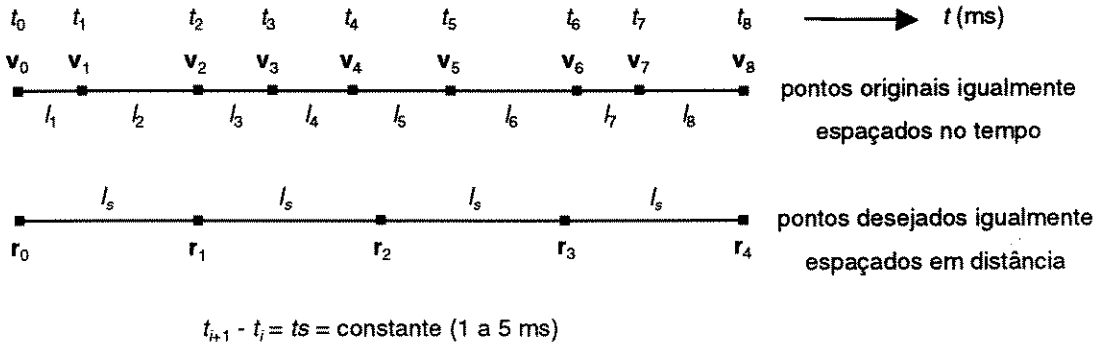


Figura 4.18: Trajetória da Fig. 4.17, esticada, e trajetória auxiliar.

Os pontos originais estão igualmente espaçados no *tempo* mas não necessariamente em distância. Assumindo que a rede neural aceita 5 quadros de entrada, a parte inferior da Fig. 4.18 mostra a mesma trajetória anterior mas com somente 5 pontos igualmente espaçados em *distância* ($\mathbf{r}_0$ a $\mathbf{r}_4$), obtidos por interpolação dos pontos (vetores) originais. Indicam-se, também, as distâncias l_s entre estes pontos.

Veremos como a técnica trace segmentation faz para calcular estes 5 pontos (ou vetores), que são os que alimentarão a rede neural. Uma condição que a análise deve cumprir é que o número de pontos $\mathbf{v}_j$ seja bem maior que o número de pontos $\mathbf{r}_i$ (é desejável que a análise seja feita a intervalos de 1 a 5 ms). O algoritmo *trace segmentation* consiste em:

1. Calcular o comprimento da trajetória original: $L_T = l_1 + l_2 + \ldots + l_8 = \sum_{k=1}^{8} l_k$.

Generalizando:
$$L_T = \sum_{k=1}^{m-1} l_k \tag{4.2}$$

onde m é o número original de vetores (isto é, o número de quadros de análise utilizado). No caso de empregar-se distância euclidiana, cada segmento l_k é obtido calculando a raiz quadrada da soma dos quadrados das diferenças entre as componentes dos vetores que estão nos extremos do segmento.

2. Calcular l_s que será a distância entre os novos pontos da curva (ou comprimento dos segmentos): $l_s = L_T / 4$, *no exemplo em questão.*

Generalizando:
$$l_s = \frac{L_T}{n-1} \tag{4.3}$$

onde n é o número de vetores desejado para representar a palavra.

3. Para cada ponto (vetor) $\mathbf{r}_i$ a calcular, encontre os pontos (vetores) $\mathbf{v}_j$ e $\mathbf{v}_{j-1}$ da trajetória original, que verificam as seguintes desigualdades:

$$i.l_s < \sum_{k=1}^{j} l_k \tag{4.4a}$$

$$i.l_s > \sum_{k=1}^{j-1} l_k \tag{4.4b}$$

Estes vetores são os vizinhos posterior e anterior do vetor $\mathbf{r}_i$ e se utilizarão para a obtenção do mesmo.

4. Calcule $\mathbf{r}_i$ interpolando linearmente $\mathbf{v}_j$ e $\mathbf{v}_{j-1}$ com a seguinte equação:

$$\mathbf{r}_i = \mathbf{v}_{j-1} + \alpha(\mathbf{v}_j - \mathbf{v}_{j-1}) \tag{4.5}$$

onde

$$\alpha = \frac{i.l_s - \sum_{k=1}^{j-1} l_k}{l_j} \tag{4.6}$$

No passo 3 do algoritmo pode ocorrer que: $i.l_s = \sum_{k=1}^{j} l_k$.

Isto significa que o vetor a calcular, $\mathbf{r}_i$, coincide com o vetor $\mathbf{v}_j$ da curva original sendo, portanto, desnecessário aplicar as Eqs. 4.5 e 4.6. Basta replicar $\mathbf{v}_j$.

Um problema do algoritmo *Trace Segmentation* é que as componentes dos vetores $\mathbf{v}_j$ com maiores variações durante a elocução da palavra, tendem a dominar as medidas de distância nas quais a segmentação é baseada. Estas variações podem ser devidas a ruídos e não a eventos foneticamente significativos, degradando a qualidade da segmentação. Para minimizar este efeito, o algoritmo *Individual Trace Segmentation* aplica o algoritmo TS a cada uma das componentes dos vetores $\mathbf{v}_j$. Desta forma, a trajetória de cada coeficiente é segmentada separadamente. Após a aplicação do algoritmo TS p vezes (uma em cada componente), obtém-se o vetor $\mathbf{r}_i$.

As equações utilizadas no algoritmo ITS são as mesmas do TS. Os segmentos l_k utilizados na Eq. 4.2, correspondem aos módulos das diferenças entre as componentes (isto, utilizando-se distância euclidiana), e na Eq. 4.5 os vetores são substituídos por suas componentes.

4.4.2 Resultados obtidos

Nas simulações realizadas os intervalos de análise testados, *ts*, foram 2 ms e 4 ms. Empregaram-se quadros de análise de 30 ms, ponderados com janelas de Hamming. Quanto ao número final de quadros (vetores) de entrada à rede neural, testou-se o uso de 40 e 60 quadros. Os resultados aqui apresentados correspondem a 40 quadros, já que com este número as taxas de erro foram menores que com 60 quadros. Isto se deve, presumivelmente, a que como com 60 quadros de entrada o número de sinapses da rede é maior, seria necessária uma seqüência de treinamento também maior para obter uma boa estimativa das sinapses. Como se utilizou a mesma seqüência de treinamento para ambos os casos, os resultados foram piores com 60

quadros. Os parâmetros utilizados como entrada da rede neural foram os mel-cepstrais (12 por quadro).

Os sinais de treinamento e teste foram sem ruído, pré-enfatizados com um fator de 0,9. As redes neurais testadas foram a backpropagation, com uma camada escondida de 30 e 50 neurônios, e a rede LVQ, com 11 e 25 neurônios.

Os resultados obtidos são mostrados a seguir. A Tabela 4.37 corresponde à aplicação da técnica TS e a 4.38 à aplicação da técnica ITS. Nestas tabelas, *"ts"* corresponde ao intervalo de tempo entre cálculo de parâmetros (2 ou 4 ms); *"h"* é o número de neurônios da camada escondida nas redes backpropagation (30 e 50), ou o número de neurônios das redes LVQ (11 e 25).

Tabela 4.37: Taxas de erro em % para os 4 vocabulários utilizando a técnica TS com redes backpropagation e LVQ. Para backpropagation, h = 30 e 50 neurônios. Para LVQ, h = 11 e 25 neurônios. ts = 2 e 4 ms.

	ts = 2 ms				ts = 4 ms			
Vocabulário	**h = 30**	**50**	**h = 11**	**25**	**h = 30**	**50**	**h = 11**	**25**
Dígitos (D)	7,19	7,40	7,81	7,92	7,19	7,50	7,81	7,81
Cálculo (C)	7,58	7,96	8,71	8,58	7,96	7,96	8,71	9,09
Movimento (M)	7,58	7,96	8,33	8,58	7,58	7,85	8,71	8,71
D+C+M	12,37	12,63	14,36	14,15	12,37	12,72	14,58	14,58

Tabela 4.38: Taxas de erro em % para os 4 vocabulários utilizando a técnica ITS com redes backpropagation e LVQ. Para backpropagation, h = 30 e 50 neurônios. Para LVQ, h = 11 e 25 neurônios. ts = 2 e 4 ms.

	ts = 2 ms				ts = 4 ms			
Vocabulário	**h = 30**	**50**	**h = 11**	**25**	**h = 30**	**50**	**h = 11**	**25**
Dígitos (D)	6,88	7,29	7,19	7,19	7,19	7,19	7,29	7,29
Cálculo (C)	7,58	7,96	7,96	8,33	7,58	7,96	8,33	8,33
Movimento (M)	7,96	7,96	8,21	8,33	8,21	8,33	8,71	8,58
D+C+M	12,15	12,50	14,19	14,19	12,15	12,63	14,45	14,58

Analisando as tabelas acima, pode ser visto que a técnica ITS apresenta resultados levemente superiores aos conseguidos com a técnica TS, mas sem chegar a superar os obtidos na Seção 4.1.1 quando se analisavam todas as palavras com igual número de quadros. Naquela seção foi visto que utilizando 40 quadros de análise por palavra e uma rede backpropagation com 30 neurônios na camada escondida, obtinham-se as seguintes taxas de erro (Tabela 4.5, reproduzida parcialmente aqui para efeitos de comparação):

Tabela 4.5 (reprodução parcial): Taxas de erro em % para os 4 vocabulários utilizando diferentes parâmetros de entrada e 40 quadros de análise por palavra. Utiliza-se uma rede backpropagation com 1 camada escondida de 30 neurônios.

Parâmetros	Dígitos (D)	Cálculo (C)	Movimento (M)	D+C+M
...	...	...	...	...
Mel (DCT)	1,88	3,03	2,65	6,64
...	...	...	...	...

Pode ver-se que as taxas de erro da Tabela 4.5 são bem menores que as obtidas com as técnicas TS e ITS, não justificando-se, portanto, sua utilização.

4.5 Uso de interpolação e dizimação de quadros para manter constante o número de entradas da rede neural

O problema principal das redes neurais em reconhecimento estático de fala é o fato das redes terem um número fixo de entradas e as palavras a reconhecer terem durações diferentes. Devido à necessidade de manter constante o número de entradas da rede neural, faz-se necessário:

1. analisar todas as palavras com igual número de quadros. Neste caso as palavras mais longas terão quadros mais longos e isto pode fazer com que a duração dos mesmos supere os 40 ou 50 ms, tornando os trechos sob análise não estacionários;
2. analisar todas as palavras com quadros do mesmo comprimento e depois:
   a. colocar o número de entradas da rede suficientemente grande de forma que caiba a palavra mais longa; para palavras mais curtas completar as entradas vazias da rede com zeros;
   b. dizimar/interpolar o número de quadros de análise de cada palavra de forma que todas fiquem com igual número de quadros.

Ao longo das Seções 4.1, 4.2 e 4.3 foram feitas simulações das opções 1 e 2.a, obtendo-se resultados melhores com a opção 1. Para testar a opção 2.b, deve-se estabelecer algum critério de dizimação (eliminação), e interpolação (incorporação) de quadros. Uma forma de dizimação/interpolação foi implementada na Seção 4.4 ao simular as técnicas TS e ITS. Nestas técnicas analisavam-se as palavras com quadros de 30 ms a uma taxa bem alta (a cada 2 ou 4 ms), calculavam-se os vetores de parâmetros correspondentes, estes vetores determinavam uma

trajetória espacial que era medida utilizando-se distância euclidiana, e sobre esta trajetória calculavam-se pontos igualmente espaçados em distância utilizando-se os vetores vizinhos aos pontos escolhidos. O número de pontos escolhidos era o número de quadros de análise desejado. Com este método não se conseguiram bons resultados. As taxas de erro resultantes foram maiores que as obtidas anteriormente. Nesta seção implementar-se-á uma forma de dizimação e interpolação diferente, e uma forma de análise do sinal de fala também diferente.

Dizimar implica em eliminar quadros da palavra analisada. Os quadros eliminados podem conter informações relevantes para o reconhecimento, dificultando-o ou até impossibilitando-o. Por outro lado, se as informações contidas nos quadros eliminados forem redundantes ou irrelevantes, o reconhecimento da palavra será pouco afetado. A seguir vamos propor critérios para a eliminação de quadros, procurando preservar a informação considerada relevante para o reconhecimento.

Uma opção lógica para a dizimação é eliminar aqueles quadros onde a variação espectral em relação aos vizinhos seja mínima. Isto porque eliminar alguns quadros de uma região espectralmente estável (um som sonoro, por exemplo), não afeta significativamente a informação discriminante. Têm mais peso para o reconhecimento as descontinuidades espectrais (transições dos sons), que a duração de uma região estável. De fato, sistemas concatenativos de conversão texto-fala, como por exemplo o que utiliza a técnica PSOLA (Charpentier & Moulines, 1989), baseiam a segmentação das unidades acústicas de síntese nesta característica (as transições, por serem regiões importantes para a inteligibilidade e qualidade da síntese, são preservadas, enquanto que a segmentação/concatenação das unidades é realizada nas regiões estáveis dos sons, onde erros são menos percebidos).

Para a interpolação dos quadros o critério lógico é o mesmo. Interpolar significa incorporar quadros inexistentes a uma palavra. Com o critério utilizado, a incorporação de quadros é implementada duplicando-se quadros de uma região espectralmente estável, uma vez que a informação incorporada diz respeito, apenas, à duração desta região. Como exemplo, poderia aumentar-se a duração de um som vozeado (região espectralmente estável), sem alterar significativamente a palavra. Analisando sentenças faladas a diferentes velocidades, verifica-se que as transições são bem menos elásticas que as regiões estáveis dos sons; as principais diferenças ocorrem nas regiões estáveis sendo as transições pouco afetadas.

Se o critério utilizado para dizimar e interpolar quadros é a estabilidade espectral dos mesmos, deve-se utilizar algum mecanismo que permita avaliar o quão estável ou instável um quadro é. Neste trabalho empregaram-se os coeficientes delta-energia e delta-mel-cepstrais para avaliar a estabilidade espectral de um quadro.

Os coeficientes delta são aproximações das derivadas em relação ao tempo dos coeficientes originais. Foram calculados utilizando-se a Eq. 3.14 do Cap. 3, empregando os 3 quadros anteriores e os 3 posteriores ao quadro em questão (Cap. 3, Seção 3.3). Quanto maiores forem os coeficientes delta de um quadro (em valor absoluto), maiores serão as diferenças espectrais do mesmo em relação aos vizinhos (descontinuidades espectrais). Estes quadros devem ser preservados. Quanto menores forem os coeficientes delta de um quadro (em valor absoluto), menores serão suas diferenças espectrais com os vizinhos. São bons candidatos para eliminar ou reproduzir.

A razão de se escolher o delta-energia e os delta-mel-cepstrais para a avaliação da estabilidade espectral dos quadros, é a verificação de que com ambos coeficientes é possível identificar a maior parte das transições que ocorrem na fala. Durante estas transições os deltas mencionados apresentam picos. Isto pode ser visto na figura seguinte, onde é mostrado o sinal de fala correspondente à palavra ***rápido***, e as curvas de delta-energia e do módulo do vetor de delta-mel-cepstrais.

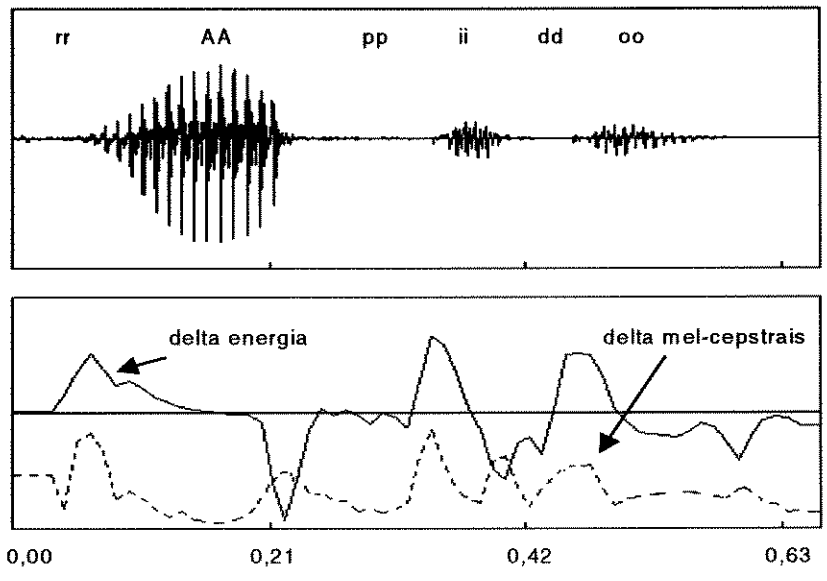


Figura 4.19: Sinal de fala da palavra RÁPIDO e curvas de delta-energia e delta-mel-cepstrais.

As transições dos sons que ocorrem na palavra ***rápido*** são: de silêncio (indicado com o sinal ##), para fricativa (##_rr), de fricativa para vogal (rr_AA), de vogal para plosiva (AA_pp), de plosiva para vogal (pp_ii), de vogal para plosiva sonora (ii_dd), de plosiva sonora para vogal (dd_oo) e de vogal para silêncio (oo_##). Perceba-se como nestas transições o delta-energia e o módulo do vetor de delta-mel-cepstrais apresentam picos.

Nem sempre as transições são colocadas em evidência pelos coeficientes delta. Portanto, a escolha de um método mais eficiente para determinar as regiões espectralmente estáveis e instáveis de uma palavra, é um tópico a ser analisado mais profundamente.

O delta final que foi adotado aqui como critério de decisão da estabilidade espectral de um quadro, foi a soma ponderada dos deltas anteriores. A ponderação se fez necessária para que ambos termos interviessem com o mesmo peso no delta final. O delta adotado foi:

$$d_T = |d_e| + k \,.\, mod\_d_{mel} \tag{4.7}$$

Nesta equação, d_T é o delta-total utilizado como critério de estabilidade, $|d_e|$ é o valor absoluto de delta-energia, mod_d_{mel} é o módulo do vetor de delta-mel-cepstrais e k é a constante de ponderação. Para k foram testados três valores: o que iguala as variâncias de $|d_e|$ e de mod_d_{mel}, o que iguala os desvios médios e o que iguala os intervalos de variação. O desvio médio (adev: *average deviation*), é um estimador mais robusto que a variância, da variabilidade dos valores ao redor da média.

As equações utilizadas para calcular as variâncias e desvios médios de delta energia e do módulo do vetor de delta-mel-cepstrais, foram as seguintes ("Numerical Recipes in C", W. Press et al.; Cambridge University Press, 1992):

$$\text{média } (x_1, \ldots, x_N) = m_x = \frac{1}{N} \sum_{j=1}^{N} x_j \tag{4.8}$$

$$\begin{aligned} \text{variância } (x_1, \ldots, x_N) = \sigma_x^2 &= \frac{1}{N-1} \sum_{j=1}^{N} (x_j - m_x)^2 \\ &\approx \frac{1}{N-1} \left\{ \sum_{j=1}^{N} (x_j - m_x)^2 - \frac{1}{N} \left[\sum_{j=1}^{N} (x_j - m_x) \right]^2 \right\} \end{aligned} \tag{4.9}$$

$$\text{adev } (x_1, \ldots, x_N) = \frac{1}{N-1} \sum_{j=1}^{N} |x_j - m_x| \,. \tag{4.10}$$

A equação utilizada para calcular a variância minimiza o "*rundoff error*", e se denomina "algoritmo de dois passos corrigido".

Uma questão que surge ao dizimar e interpolar quadros é se devem-se dizimar/interpolar os quadros com menor delta independentemente de suas posições na palavra, ou se deve-se limitar o número de quadros consecutivos que podem ser eliminados ou duplicados em uma região. Foi testada a primeira opção e obtiveram-se bons resultados. Para a segunda opção dividiu-se a palavra em várias regiões (entre 2 e 8), com aproximadamente o mesmo número de quadros em todas elas, eliminando/duplicando um quadro de cada região por vez. Apenas quando em todas as regiões tinham-se eliminado/duplicado quadros, é que uma região era candidata a nova eliminação/duplicação de quadros. Os resultados foram melhores empregando-se a primeira opção (isto é, não colocando limites ao número de quadros consecutivos que podem eliminar-se/duplicar-se).

Uma outra questão que surge, desta vez quando são utilizados parâmetros delta como entradas da rede neural, é a seguinte: os parâmetros delta calculados na palavra original representam as variações espectrais em relação aos quadros vizinhos verdadeiros. Depois de feita a dizimação, os deltas recalculados representam variações espectrais entre quadros não vizinhos. Se houve um quadro eliminado espera-se que o novo delta tenha um valor maior que o original (não necessariamente duplo), uma vez que a correlação entre quadros originalmente mais afastados tende a ser menor. Neste caso, devem ser ponderados os novos deltas para levar em conta a dizimação que houve ou devem utilizar-se os deltas originais? Com a interpolação o questionamento é similar.

Nas próximas duas seções apresentar-se-ão os resultados obtidos ao se aplicar estas técnicas de dizimação e interpolação, em palavras analisadas com quadros de comprimento fixo e em palavras onde se realizou análise síncrona com o pitch. Veremos que com ambos tipos de análise e a técnica de dizimação/interpolação de quadros proposta se obtiveram os melhores resultados da tese, sendo que com análise síncrona com o pitch os resultados foram superiores. As técnicas foram testadas com sinais sem ruído e com sinais ruidosos.

4.5.1 Análise com quadros de comprimento fixo

Tanto na análise com quadros de comprimento fixo como na análise síncrona com o pitch, utilizaram-se janelas de Hanning. A razão de se escolher esta janela é que a amplitude em suas bordas é

zero, sendo esta uma característica desejável na análise síncrona com o pitch. No caso da análise com quadros de comprimento fixo esta condição não é necessária. Neste caso utilizou-se a janela de Hanning para poder comparar os resultados obtidos com os da análise síncrona com o pitch.

Os quadros de análise empregados foram de 20 ms a cada 10 ms. Os sinais de voz originais foram pré-enfatizados com um fator igual a 0,9.

Após a análise aplicaram-se as técnicas de dizimação e interpolação de quadros explicadas na seção anterior, de forma que todas as palavras ficassem com igual número de quadros. Os sinais dizimados/interpolados foram alimentados a uma rede neural do tipo perceptron multicamadas (backpropagation), com uma camada escondida contendo 30 ou 50 neurônios para os vocabulários de dígitos, cálculo e movimento, e 50 ou 70 neurônios para o vocabulário completo. O número de neurônios de saída da rede foi feito igual ao número de palavras a reconhecer. O número de entradas foi feito igual ao produto do número de quadros da palavra após a dizimação/interpolação, pelo número de coeficientes calculados em cada quadro.

Em uma primeira fase utilizou-se, apenas, o vocabulário de dígitos, coeficientes mel-cepstrais (12 por quadro), e 30 neurônios na camada escondida da rede neural, com o objetivo de definir: 1) o número final de quadros a utilizar (40, 50, 60 ou 70 quadros), 2) o melhor fator de ponderação no cálculo do delta total (desvio médio, variância ou intervalo de variação), e 3) o número de regiões de dizimação e interpolação a empregar, caso se limitasse o número de quadros consecutivos que poderiam ser eliminados/duplicados (para este ponto foram testadas 1, 2, 4, 6 e 8 regiões).

Os principais resultados obtidos nesta primeira fase são mostrados a seguir (Tabela 4.39). Nesta tabela, **numQ** indica o número final de quadros utilizado, **r** o número de regiões de dizimação/interpolação empregado, e **adev** (desvio médio), **var** (variância) e **range** (intervalo de variação), o fator de ponderação utilizado no cálculo do delta total.

Tabela 4.39: Taxas de erro em % para os dígitos, utilizando dizimação e interpolação de quadros e análise com janelas de Hanning de 20 ms a cada 10 ms. Rede neural utilizada: backpropagation com uma camada escondida de 30 neurônios. Parâmetros de entrada da rede: coeficientes mel-cepstrais.

	adev			var			range		
numQ	**r=1**	**r=4**	**r=8**	**r=1**	**r=4**	**r=8**	**r=1**	**r=4**	**r=8**
40	1,56	3,44	3,85	3,65	6,56	6,88	1,67	3,75	3,85
50	0,94	3,75	3,75	3,44	6,88	7,19	0,94	3,44	3,75
60	1,25	3,85	4,06	3,75	6,88	7,19	1,25	3,65	3,75
70	2,19	4,06	4,06	4,38	7,81	7,92	2,19	4,06	4,69

Como pode ser visto, os melhores resultados se obtiveram dizimando ou interpolando quadros até se alcançar um total de 50 quadros/palavra. Quanto ao número de regiões de dizimação/interpolação a usar, o emprego de mais de uma região degradou a informação discriminante causando um aumento nas taxas de erro do sistema. Assim, não colocar limites ao número consecutivo de quadros a eliminar ou duplicar (r= 1), causou os melhores resultados. Finalmente, quanto ao fator de ponderação a utilizar no cálculo do delta total, o emprego de *adev* ou *range* produziu praticamente os mesmos resultados. A utilização da *variância*, no entanto, aumentou as taxas de erro sensivelmente, indicando que não é um bom parâmetro para ser usado na estimação das descontinuidades espectrais da fala com o método aqui proposto.

Baseados nestes resultados, nas simulações seguintes utilizaram-se, apenas, *adev* e *range* como fator de ponderação no delta total, 50 quadros de análise por palavra –como objetivo da dizimação/interpolação– e número de quadros consecutivos a eliminar ou duplicar sem limite.

As tabelas seguintes, 4.40 a 4.43, mostram os resultados obtidos com este método nos 4 vocabulários sob teste (dígitos, comandos de cálculo, comandos de movimento e vocabulário completo).

Foram testados 30 e 50 neurônios escondidos para os três primeiros vocabulários, e 50 e 70 neurônios escondidos para o vocabulário completo, coeficientes mel-cepstrais e mel-cepstrais mais energia como entradas da rede neural, e adev e range no cálculo do delta total. Empregou-se, como mencionado, numQ = 50 (numQ: número de quadros após a dizimação/interpolação).

Tabela 4.40: Taxas de erro para os dígitos utilizando dizimação e interpolação de quadros e análise com janelas de Hanning de 20 ms a cada 10 ms. Rede neural: backpropagation com uma camada escondida de 30 ou 50 neurônios (h). Número de quadros consecutivos a eliminar/duplicar: ilimitado. Parâmetros de entrada: mel-cepstrais e mel-cepstrais mais energia (E). numQ = 50.

Vocabulário	h	Parâmetros	Fator Pond.	Erro %
Dígitos (D)	30	mel	adev	0,94
			range	0,94
		mel+E	adev	1,25
			range	1,52
	50	mel	adev	0,94
			range	1,25
		mel+E	adev	1,88
			range	1,88

Tabela 4.41: Taxas de erro para os comandos de cálculo, utilizando dizimação e interpolação de quadros e análise com janelas de Hanning de 20 ms a cada 10 ms. Rede neural: backpropagation com uma camada escondida de 30 ou 50 neurônios (h). Número de quadros consecutivos a eliminar/duplicar: ilimitado. Parâmetros de entrada: mel-cepstrais e mel-cepstrais mais energia (E). numQ = 50.

Vocabulário	h	Parâmetros	Fator Pond.	Erro %
Cálculo (C)	30	mel	adev	2,14
			range	2,27
		mel+E	adev	2,40
			range	2,65
	50	mel	adev	2,27
			range	2,65
		mel+E	adev	3,16
			range	3,16

Tabela 4.42: Taxas de erro para os comandos de movimento, utilizando dizimação e interpolação de quadros e análise com janelas de Hanning de 20 ms a cada 10 ms. Rede neural: backpropagation com uma camada escondida de 30 ou 50 neurônios (h). Número de quadros consecutivos a eliminar/duplicar: ilimitado. Parâmetros de entrada: mel-cepstrais e mel-cepstrais mais energia (E). numQ = 50.

Vocabulário	h	Parâmetros	Fator Pond.	Erro %
Movimento (M)	30	mel	adev	1,52
			range	1,64
		mel+E	adev	1,52
			range	1,89
	50	mel	adev	2,27
			range	2,27
		mel+E	adev	2,65
			range	2,90

Tabela 4.43: Taxas de erro em % para o vocabulário completo, utilizando dizimação e interpolação de quadros e análise com janelas de Hanning de 20 ms a cada 10 ms. Rede neural: backpropagation com uma camada escondida de 50 ou 70 neurônios (h). Número de quadros consecutivos a eliminar/duplicar: ilimitado. Parâmetros de entrada: mel-cepstrais e mel-cepstrais mais energia (E). numQ = 50.

Vocabulário	h	Parâmetros	Fator Pond.	Erro %
D+C+M	50	mel	adev	5,99
			range	5,86
		mel+E	adev	6,51
			range	6,42
	70	mel	adev	4,43
			range	4,52
		mel+E	adev	6,29
			range	6,38

Analisando-se as tabelas acima podemos concluir que os melhores resultados se obtiveram utilizando como entradas da rede neural os coeficientes mel-cepstrais, e como fator de ponderação no cálculo do delta total, o desvio médio *adev*. Para os dígitos, comandos de cálculo e comandos de movimento, o emprego de 30 neurônios na camada escondida produziu os melhores resultados.

Com 50 neurônios os resultados foram piores devido, provavelmente, ao tamanho limitado da seqüência de treinamento. Para o vocabulário completo, entretanto, os melhores resultados se obtiveram com 70 neurônios na camada escondida. Neste caso a seqüência de treinamento foi bem maior.

A tabela seguinte, 4.44, mostra um resumo dos resultados que tinham sido obtidos nas seções anteriores, sem se aplicar as técnicas de dizimação e interpolação de quadros. São mostradas as taxas percentuais de erro do sistema utilizando-se análise com 40 quadros por palavra e análise com quadros de 30 ms a cada 20 ms (Seções 4.1.1 e 4.1.2 respectivamente). As redes neurais empregadas foram as backpropagation com uma camada escondida de 30 neurônios para os primeiros três vocabulários, e 50 neurônios para o vocabulário completo. Parâmetros de entrada da rede: mel-cepstrais e mel-cepstrais mais energia.

Tabela 4.44: Taxas de erro em % para os 4 vocabulários, utilizando redes backpropagation com uma camada escondida de 30 neurônios (dígitos, cálculo e movimento), e 50 neurônios (vocabulário completo); análise com 40 quadros por palavra e com quadros de 30 ms recalculados a cada 20 ms. Origem: Tabelas 4.4, 4.5, 4.11 e 4.12.

	Análise com 40 quadros/palavra		**Análise com quadros de 30 ms**	
Vocabulário	mel	mel+E	mel	mel+E
Dígitos (D)	1,88	2,19	2,19	1,88
Cálculo (C)	3,03	2,65	3,41	3,03
Movimento (M)	2,65	2,65	4,92	3,79
D+C+M	6,51	6,77	8,72	8,33

Pode ser visto que a utilização de dizimação e interpolação de quadros com a técnica proposta, fez cair as taxas de erro sensivelmente para os 4 vocabulários. Tomando como referência os melhores resultados da Tabela 4.44, as quedas nas taxas de erro vão de 19,25% (vocabulário de cálculo), até 50% (vocabulário de dígitos). Para o vocabulário dos comandos de movimento a queda foi de aproximadamente 43% e para o vocabulário completo de 32%. Estes resultados são bem significativos e dizem da potencialidade da técnica. O custo a pagar pelo aumento na taxa de acerto, é um aumento da carga computacional e do tempo de pré-processamento gasto com o sinal para realizar a dizimação e interpolação dos quadros. Este aumento no tempo de pré-processamento é de aproximadamente 40%.

Na tabela seguinte, 4.45, são mostradas as taxas de erro com sinais ruidosos. Neste caso as redes foram treinadas com uma mistura de sinais sem ruído e com ruído (para estes últimos, SNR= 20 dB e 10 dB). Os testes foram feitos com os três tipos de sinais. Os resultados correspondem à utilização de 30 neurônios escondidos nos vocabulários de dígitos, cálculo e movimento, e 70 neurônios

escondidos no vocabulário completo. O fator de ponderação empregado no cálculo do delta total foi o desvio médio (adev). O número de quadros após a dizimação/interpolação foi 50.

Tabela 4.45: Taxas de erro em % para os 4 vocabulários, utilizando dizimação e interpolação de quadros e análise com janelas de Hanning de 20 ms a cada 10 ms. Treinamento e teste com sinais com e sem ruído (s/r: sem ruído). Rede neural: backpropagation com uma camada escondida de 30 neurônios (dígitos, cálculo e movimento), e 70 neurônios (vocabulário completo). Fator de ponderação no delta total: adev. Parâmetros de entrada: mel-cepstrais. numQ = 50.

Vocabulário	Conj. Teste s/r	Conj. Teste 20 dB	Conj. Teste 10 dB	Média
Dígitos (D)	1,88	2,19	3,13	2,40
Cálculo (C)	2,27	2,65	2,52	2,48
Movimento (M)	2,02	2,27	2,65	2,31
D+C+M	8,85	8,07	8,59	8,50

Como era esperado, o desempenho das redes com sinais ruidosos é degradado, mas o fato de treiná-las com os três tipos de sinal (sem ruído e com ruído de 20 e 10 dB), faz que o comportamento com estes três tipos de sinal seja bastante uniforme. Adicionalmente, o desempenho destas redes em ambientes ruidosos aplicando-se as técnicas de dizimação e interpolação propostas, também foi superior aos desempenhos obtidos quando estas técnicas não foram aplicadas (Seção 4.1.4, Tabela 4.17).

4.5.2 Análise síncrona com o pitch

Até aqui, e ao longo da tese, a análise realizada no sinal de fala para extrair parâmetros temporais e/ou espectrais, foi feita com igual número de quadros em todas as palavras ou com quadros de comprimento fixo. Este tipo de análise é *assíncrona* com o pitch. Diz-se que é *assíncrona com o pitch*, porque os quadros de análise caem em qualquer posição dentro da palavra, independentemente da posição das marcas de pitch dos trechos periódicos do sinal. Este tipo de análise apresenta dois problemas: o primeiro está relacionado à duração dos quadros de análise; o segundo à posição em que os quadros de análise caem dentro da palavra analisada.

Em relação ao primeiro problema –a duração dos quadros de análise–, usualmente escolhem-se quadros de 5 a 40 ms. Esta duração garante duas condições necessárias à análise: a primeira é que no trecho sob análise o sinal de fala possa ser considerado estacionário (na realidade varia lentamente com o tempo); o segundo é que exista um bom compromisso entre a resolução *temporal* e a resolução em *freqüência* na análise realizada. Quanto menor for a duração do quadro, maior será a

resolução temporal e menor a resolução em freqüência. Quanto maior for a duração do quadro, maior será a resolução em freqüência e menor a resolução temporal. Sendo ambas necessárias há que se chegar a um compromisso.

Vejamos o caso do cálculo do perfil de energia: quando o quadro de análise é pequeno, da ordem de um período de pitch ou menor, a energia calculada variará rapidamente acompanhando os detalhes da forma de onda no tempo. Se a duração do quadro é grande, da ordem de vários períodos de pitch, a energia calculada variará lentamente não refletindo adequadamente as variações do sinal de fala. A informação das transições abruptas, como as de uma plosiva, por exemplo, perder-se-ão. Não há um valor de quadro que satisfaça todas as situações devido a que o período de pitch varia de aproximadamente 2 ms (500 Hz) em uma criança, a 12 ms (~80 Hz) em uma voz masculina grave. Portanto, um quadro adequado para um caso não o será para outro.

No cálculo do espectro do sinal de voz ocorre a mesma coisa. Vamos supor que é analisado um trecho de sinal periódico. Se se escolhem quadros de curta duração (de um período de pitch ou menores), o sinal de fala contido no quadro de análise não apresentará periodicidades. Se, pelo contrário, escolhem-se quadros que incluam vários períodos de pitch, a resolução em freqüência do espectro será alta mas a resolução temporal será baixa (o sinal no quadro poderá ser considerado não-estacionário). Neste caso o espectro obtido será o espectro médio de um sinal variante, perdendo-se a informação das variações rápidas.

Resumindo: uma boa resolução temporal exige quadros curtos, enquanto que uma boa resolução em freqüência requer quadros longos. Com a análise assíncrona, estas duas condições nem sempre são alcançadas.

O segundo problema da análise assíncrona diz respeito à distorção que pode ser introduzida devido aos quadros de análise caírem em qualquer posição. Os quadros de análise são ponderados com algum tipo de janela (em nosso caso, Hanning). Se a região de decaimento da janela cai em cima de um pico de energia, o pico resultará atenuado. Da mesma forma uma região onde o sinal esteja caindo pode resultar em uma região estável, caso seja ponderada pela região de crescimento da janela. O cálculo do espectro, ou de qualquer parâmetro, é influenciado pela posição da janela. Se esta fica centrada em uma marca de pitch, o cálculo dará um resultado diferente de quando a janela fica centrada em um vale entre duas marcas de pitch ou em outra posição intermediária. Assume-se, aqui, que as marcas de pitch são posicionadas nos picos positivos dos segmentos sonoros.

A análise *síncrona* com o pitch minimiza os dois problemas. Esta análise consiste em centrar as janelas nas marcas de pitch, e fazer que a duração das janelas seja de dois períodos de pitch (a janela começa na marca anterior e acaba na posterior). Isto é possível nos trechos onde o sinal é periódico. Nos trechos onde o sinal não é periódico colocam-se, arbitrariamente, marcas de pitch a cada 10 ms e procede-se como nas regiões periódicas. Perceba-se que, deste modo, a análise fica casada com as características do sinal de fala. Primeiro, a duração das janelas abrange sempre dois períodos de pitch independentemente do valor deste. Isto permite obter um bom compromisso entre a resolução temporal e em freqüência da análise. Segundo, a posição das janelas minimiza a distorção do sinal para o cálculo dos parâmetros.

Devido a que não necessariamente as marcas de pitch anterior e posterior são eqüidistantes àquela onde a janela está centrada, pode ocorrer que uma janela simétrica e centrada em uma marca de pitch comece ou acabe fora das marcas vizinhas. Uma opção, neste caso, é centrar a janela na marca de pitch correspondente e fazê-la assimétrica (que chegue exatamente até as marcas anterior e posterior). Outra opção, adotada aqui, é fazer a janela simétrica começando na marca anterior e acabando na posterior, mas sem garantir que fique centrada na marca de pitch correspondente. Visto que a região central de uma janela é aproximadamente constante, esta solução não invalida a análise síncrona.

A Figura 4.20, mostra um exemplo da *análise síncrona com o pitch* realizada aqui. Nesta figura, as distâncias entre os picos positivos de energia (assinalados com setas cheias), e o centro das janelas (assinalados com setas vazias), são indicadas como Δ_i. Quando as marcas de pitch são eqüidistantes àquela onde a janela está centrada, $\Delta_i = 0$.

O problema de usar análise síncrona com o pitch, é a necessidade de se possuir um excelente detector (ou marcador) de pitch. Não interessa o *valor* do pitch senão que o detector reconheça as periodicidades no sinal de fala e coloque as marcas de pitch correspondentes nas posições corretas. O algoritmo aqui utilizado como marcador de pitch, é uma modificação do algoritmo de Kurt-Shäfer-Vincent (1983).

Visto que a análise síncrona com o pitch resulta em números diferentes de quadros de análise em função da duração e periodicidades da palavra analisada, faz-se necessário, após a análise, fazer uma dizimação/interpolação de quadros para manter constante o número de entradas da rede neural.

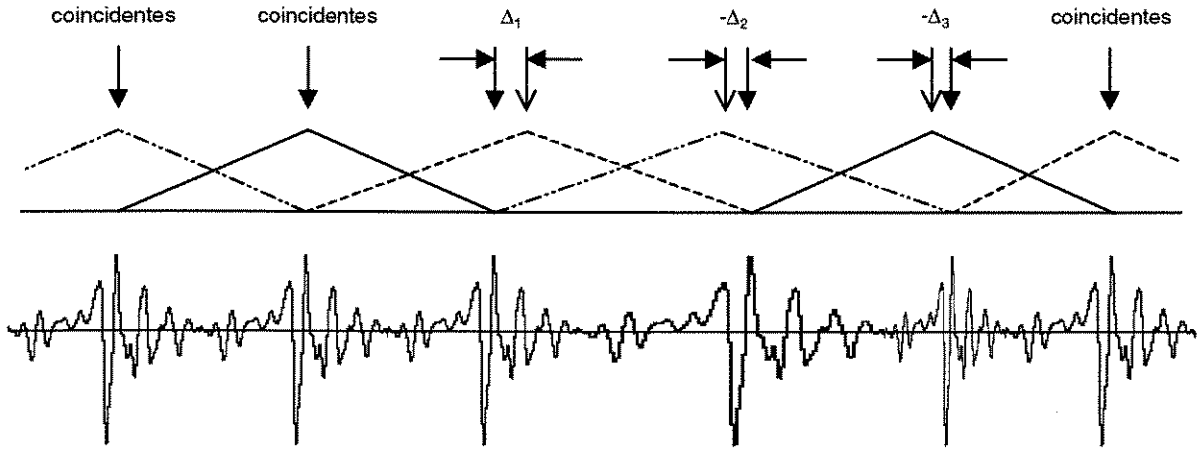


Figura 4.20: Exemplo de análise síncrona com o pitch. A parte inferior da figura mostra um sinal periódico de fala, enquanto que a parte superior mostra as janelas de análise a utilizar (no exemplo, janelas triangulares). As setas cheias indicam os picos de energia do sinal de fala; as setas vazias indicam a posição das janelas de análise.

Foi visto que o critério utilizado para dizimar/interpolar quadros é o da estabilidade espectral medida pelos parâmetros delta-energia e delta-mel-cepstrais. Dado que dentro de uma mesma palavra há quadros de duração diferente, no cálculo da energia deve-se levar em conta este fato dividindo-se a energia da janela pelo número de amostras que a janela contém.

Quanto ao tipo de janela a utilizar, quanto menor for a influência das energias vizinhas à janela de análise, mais precisos serão os parâmetros obtidos. Por isso optou-se por utilizar a janela de Hann (Hanning na literatura), que se carateriza por ter valor zero nas suas bordas. Sua equação é (Cap.3, Eq. 3.2):

$$w(n) = \begin{cases} \frac{1}{2}\left[1 - \cos\left(\frac{2\pi n}{N-1}\right)\right] & , \quad 0 \le n \le N-1 \\ 0 & , \text{ para os outros } n \end{cases} \tag{4.11}$$

Na janela de Hamming, tradicionalmente utilizada, as energias vizinhas não são zeradas já que o valor da janela nas bordas é 8% do seu valor máximo. Sua equação é (Cap. 3, Eq. 3.1):

$$w(n) = \begin{cases} 0{,}54 - 0{,}46\ \cos\left(\frac{2\pi n}{N-1}\right) & , \quad 0 \le n \le N-1 \\ 0 & , \quad \text{para os outros } n \end{cases} \tag{4.12}$$

As tabelas seguintes mostram os resultados obtidos utilizando-se análise síncrona com o pitch. Como no caso da análise com quadros de 20 ms recalculados a cada 10 ms, foram testados: número de regiões de dizimação e interpolação (1, 2, 4, 6 e 8), número final de quadros após a dizimação/interpolação (40, 50, 60 e 70 quadros), e fator de ponderação utilizado no cálculo do delta total (desvio médio *adev*, variância *var*, e intervalo de variação *range*). Em uma primeira fase utilizou-se apenas o vocabulário de dígitos, parâmetros mel-cepstrais (12 por quadro) e uma rede backpropagation com uma camada escondida de 30 neurônios, para determinar: 1) o número final de quadros a utilizar, 2) o melhor fator de ponderação no cálculo do delta total, e 3) o número de regiões de dizimação/interpolação a empregar. Os resultados mais importantes destas primeiras simulações são mostrados a seguir.

Tabela 4.46: Taxas de erro em % para os dígitos, utilizando dizimação e interpolação de quadros e análise síncrona com o pitch. Rede neural utilizada: backpropagation com uma camada escondida de 30 neurônios. Parâmetros de entrada da rede: coeficientes mel-cepstrais. **numQ** indica o número final de quadros e **r** o número de regiões de dizimação/interpolação.

	adev			var			range		
numQ	r=1	r=4	r=8	r=1	r=4	r=8	r=1	r=4	r=8
40	0,94	2,81	3,13	4,06	4,69	5,00	2,19	3,75	4,79
50	0,31	2,50	2,81	3,44	4,38	4,38	0,94	2,81	3,13
60	0,94	2,81	3,13	3,75	4,69	4,59	1,88	3,13	3,13
70	1,56	3,44	3,44	4,38	5,63	5,63	2,81	3,44	4,69

Como no caso da análise com quadros de 20 ms a cada 10 ms, os melhores resultados se obtiveram dizimando ou interpolando quadros até se alcançar um total de 50 quadros/palavra e empregando, apenas, 1 região de dizimação/interpolação. Utilizando mais de uma região (isto é, colocando limites ao número de quadros consecutivos que podem ser eliminados ou duplicados), aumentaram as taxas de erro do sistema. Quanto ao fator de ponderação a utilizar no cálculo do delta total, *adev* e *range* produziram praticamente os mesmos resultados. A utilização da *variância* aumentou as taxas de erro sensivelmente, não sendo –portanto– um bom parâmetro para estimar as descontinuidades espectrais da fala com o método aqui proposto.

Baseados nestes resultados, nas simulações seguintes utilizaram-se, apenas, *adev* e *range* como fator de ponderação no delta total, 50 quadros por palavra como objetivo da dizimação/interpolação e número ilimitado de quadros consecutivos a eliminar ou duplicar.

As tabelas seguintes, 4.47 a 4.50, mostram os resultados obtidos com esta técnica nos 4 vocabulários sob teste (dígitos, comandos de cálculo, comandos de movimento e vocabulário

completo). Foram testados 30 e 50 neurônios escondidos para os três primeiros vocabulários e 50 e 70 neurônios escondidos para o vocabulário completo, coeficientes mel-cepstrais e mel-cepstrais mais energia, e adev e range no cálculo do delta total.

Tabela 4.47: Taxas de erro para os dígitos, utilizando dizimação e interpolação de quadros e análise síncrona com o pitch. Rede neural: backpropagation com uma camada escondida de 30 ou 50 neurônios (h). Número de quadros consecutivos a eliminar/duplicar: ilimitado. Parâmetros de entrada: mel-cepstrais e mel-cepstrais mais energia (E). numQ = 50.

Vocabulário	h	Parâmetros	Fator Pond.	Erro %
Dígitos (D)	30	mel	adev	0,31
			range	0,94
		mel+E	adev	1,25
			range	1,25
	50	mel	adev	0,62
			range	1,25
		mel+E	adev	0,62
			range	1,25

Tabela 4.48: Taxas de erro para os comandos de cálculo, utilizando dizimação e interpolação de quadros e análise síncrona com o pitch. Rede neural: backpropagation com uma camada escondida de 30 ou 50 neurônios (h). Número de quadros consecutivos a eliminar/duplicar: ilimitado. Parâmetros de entrada: mel-cepstrais e mel-cepstrais mais energia (E). numQ = 50.

Vocabulário	h	Parâmetros	Fator Pond.	Erro %
Cálculo (C)	30	mel	adev	1,89
			range	2,02
		mel+E	adev	2,65
			range	2,65
	50	mel	adev	2,65
			range	2,65
		mel+E	adev	3,03
			range	3,03

Tabela 4.49: Taxas de erro para os comandos de movimento, utilizando dizimação e interpolação de quadros e análise síncrona com o pitch. Rede neural: backpropagation com uma camada escondida de 30 ou 50 neurônios (h). Número de quadros consecutivos a eliminar/duplicar: ilimitado. Parâmetros de entrada: mel-cepstrais e mel-cepstrais mais energia (E). numQ = 50.

Vocabulário	h	Parâmetros	Fator Pond.	Erro %
Movimento (M)	30	mel	adev	1,14
			range	1,52
		mel+E	adev	1,52
			range	1,89
	50	mel	adev	1,89
			range	2,27
		mel+E	adev	1,89
			range	2,65

Tabela 4.50: Taxas de erro em % para o vocabulário completo, utilizando dizimação e interpolação de quadros e análise síncrona com o pitch. Rede neural: backpropagation com uma camada escondida de 50 ou 70 neurônios (h). Número de quadros consecutivos a eliminar/duplicar: ilimitado. Parâmetros de entrada: mel-cepstrais e mel-cepstrais mais energia (E). numQ = 50.

Vocabulário	h	Parâmetros	Fator Pond.	Erro %
D+C+M	50	mel	adev	5,86
			range	5,73
		mel+E	adev	6,51
			range	6,51
	70	mel	adev	4,30
			range	4,30
		mel+E	adev	5,86
			range	6,12

Analisando as tabelas acima podemos concluir que os melhores resultados se obtiveram utilizando coeficientes mel-cepstrais e desvio médio (adev) como fator de ponderação no cálculo do delta total. Para os dígitos, comandos de cálculo e comandos de movimento, o emprego de 30 neurônios na camada escondida produziu os melhores resultados, enquanto que para o vocabulário completo os melhores resultados se obtiveram com 70 neurônios.

Comparando os resultados obtidos aqui com os da análise com quadros de comprimento fixo (Tabelas 4.40 a 4.43), a análise síncrona com o pitch apresentou melhores resultados. Se a comparação é feita com os resultados obtidos sem se aplicar a técnica de dizimação e interpolação de quadros (Tabela 4.44), as diferenças são ainda maiores. As quedas nas taxas de erro vão de 29% (comandos de cálculo) a 84% (dígitos). Este último caso implica em errar 84% menos. Para os comandos de movimento a queda foi de 57% e para o vocabulário completo foi de 34%. Com quadros de 20 ms a cada 10 ms tinham se obtido quedas de 19% a 50%. Quanto aos tempos de processamento, a análise síncrona com o pitch é bem mais demorada que a análise com quadros fixos. O acréscimo de tempo em relação a esta última é de mais de 60%. Some-se a isto o tempo gasto na dizimação e interpolação dos quadros (aproximadamente o mesmo nos dois tipos de análise).

Para completar as simulações desta seção, os sistemas foram testados com sinais ruidosos. Na tabela seguinte, 4.51, são mostrados os resultados obtidos. As redes foram treinadas com uma mistura de sinais com e sem ruído (para sinais ruidosos, SNR= 20 dB e 10 dB). Os testes foram feitos com os três tipos de sinais. Os resultados correspondem à utilização de 30 neurônios escondidos nos vocabulários de dígitos, cálculo e movimento, e 70 neurônios escondidos no vocabulário completo. O fator de ponderação empregado no cálculo do delta total foi o desvio médio (adev). O número de quadros após a dizimação/interpolação foi 50.

Tabela 4.51: Taxas de erro em % para os 4 vocabulários, utilizando dizimação e interpolação de quadros e análise síncrona com o pitch. Treinamento e teste com sinais com e sem ruído (s/r: sem ruído). Rede neural: backpropagation com uma camada escondida de 30 neurônios (dígitos, cálculo e movimento), e 70 neurônios (vocabulário completo). Fator de ponderação no delta total: adev. Parâmetros de entrada: mel-cepstrais. numQ = 50.

Vocabulário	Conj. Teste s/r	Conj. Teste 20 dB	Conj. Teste 10 dB	Média
Dígitos (D)	1,77	2,19	3,02	2,33
Cálculo (C)	2,27	2,52	2,52	2,44
Movimento (M)	2,27	2,27	2,52	2,35
D+C+M	8,33	8,07	8,46	8,29

Como era de se esperar, o desempenho das redes com sinais ruidosos piorou, mas se manteve superior aos desempenhos obtidos quando a dizimação e interpolação de quadros não foi aplicada (Seção 4.1.4).

4.5.3 Conclusões

Podemos concluir que a técnica de dizimação e interpolação de quadros proposta, é superior, desde o ponto de vista de taxas de acerto da rede, a quaisquer das técnicas apresentadas antes. Seja utilizando quadros de comprimento fixo ou análise síncrona com o pitch,, a dizimação e interpolação de quadros permite obter resultados melhores que os obtidos com a análise com igual número de quadros para todas as palavras, ou com quadros de comprimento fixo e completando as entradas vazias da rede com zeros. As melhoras no desempenho do sistema de reconhecimento são sensíveis, diminuindo as taxas de erro entre 19% e 84%. O custo computacional destas melhoras é elevado. Devem calcular-se parâmetros delta energia e delta mel-cepstrais em todos os quadros, com eles computar-se o delta total do quadro utilizando a ponderação devida, depois devem-se dizimar ou interpolar quadros até se chegar ao número desejado e então alimentar-se a rede neural com os parâmetros correspondentes. No caso da análise síncrona com o pitch, há ainda o custo adicional de colocar marcas de pitch no sinal de fala e centrar as janelas de análise nelas. Neste caso a necessidade de um bom marcador de pitch é primordial para o sucesso da análise. Em ambos casos (análise síncrona com o pitch ou com quadros de comprimento fixo), o processamento só pode ser realizado após ter chegado a palavra completa, devido à necessidade de se definir quantos quadros devem ser eliminados ou duplicados. O ganho no resultado, porém, é significativo.

Capítulo 5

Adaptação ao locutor

5.1 Introdução

Os sistemas de reconhecimento de fala independentes do locutor apresentam usualmente taxas de acerto menores que os sistemas dependentes do locutor.

Nos sistemas dependentes do locutor a rede é treinada, apenas, com a voz do locutor que a utilizará, enquanto que nos sistemas independentes do locutor, a rede deve ser treinada com o maior número possível de locutores. Em ambos os casos, tratando-se de reconhecimento de palavras isoladas, a rede constrói padrões que correspondem às médias das diferentes formas de pronunciar as palavras que o sistema deve reconhecer. A informação que a rede aprende inclui características espectrais da voz do falante, já que ela é alimentada com parâmetros espectrais e temporais extraídos da palavra falada.

No caso de existir apenas um locutor, este aprendizado é mais simples. Não existe tanta variedade na forma de pronunciar palavras por uma mesma pessoa. No caso de sistemas independentes do locutor existem inúmeros locutores e, consequentemente, inúmeras formas de falar. Neste caso o aprendizado é mais complicado; os padrões que a rede constrói correspondem às médias das diferentes formas de pronunciar uma palavra pelos diferentes locutores. Como existem inúmeros locutores e inúmeras formas de realizar esta pronuncia, quanto maior for o número de exemplos utilizado para treinar a rede neural mais acurada será a média. A pesar de isto tornar eficiente o sistema para o seu uso com um número grande de locutores, o desempenho fica degradado quando um único locutor faz uso do sistema. Isto ocorre devido a que dificilmente as características intrínsecas de um locutor coincidem com as médias que a rede aprendeu. Pode ocorrer, ainda, que as características de um determinado locutor estejam

tão longe da média aprendida pela rede, que a taxa de acerto do sistema caia consideravelmente quando usado por aquele locutor.

Neste capítulo apresentar-se-á o método por nós proposto para adaptar o sistema de reconhecimento de fala às características espectrais da voz do locutor, visando aumentar as taxas de acerto do sistema. O objetivo é fazer a adaptação sem necessidade de retreinar a rede neural com cada novo locutor, assim como não despender muito tempo na adaptação do sistema. Duas alternativas foram testadas com este fim: separação dos locutores em vozes masculinas e femininas, e agrupamento de locutores com vozes espectralmente similares, de forma automática.

Como dito, o sistema tenta adaptar-se às *características espectrais* da voz do locutor e não a sua *forma de falar* (sotaque). Quanto à forma de falar, a acuidade do sistema dependerá da riqueza da base de dados com que foi treinado.

5.2 Análise das características espectrais da voz do locutor

Segundo a literatura, um fator que influencia fortemente nas características espectrais da voz das pessoas é o comprimento do trato vocal (*vocal tract length*). É claro que outros fatores, tais como a forma do trato vocal e o estado emocional de quem fala, também influenciam nas características espectrais da voz. Fant (1973) mostrou que quanto mais comprido é o trato vocal mais baixa é a posição dos formantes no espectro e vice-versa. Esta relação não é puramente linear, dependendo do contexto. Fisicamente as mulheres possuem um trato vocal menor e por isso seus formantes estão deslocados para cima no espectro. O mesmo acontece com crianças; onde o trato vocal ainda não acabou de se desenvolver.

Vários estudos e propostas têm sido feitos para eliminar as diferenças em *vocal tract length* (Kamm et al., 1995, Zhan and Waibel, 1997, Lee and Rose, 1996, Eide and Gish, 1996, etc.). As técnicas propostas visam normalizar de alguma forma o comprimento do trato vocal, com o objetivo de diminuir sua influência nos parâmetros espectrais extraídos da fala. Esta normalização denomina-se, também, *normalização do locutor*.

Em sistemas baseados em redes neurais, a adaptação ao locutor pode ser feita – entre outras formas – mapeando-se o espaço de características espectrais antes da entrada à rede neural, mapeando-se as saídas da rede, ou normalizando-se o espectro do locutor para um espectro genérico ou médio.

Duas técnicas foram propostas para normalizar o espectro de um locutor: *frequency warping* e *Bark/Mel scale warping*. Na primeira técnica deforma-se o eixo de freqüências com uma regra linear ou exponencial, de forma que as posições dos formantes de um locutor específico estejam próximas das posições dos formantes de um espectro médio. Na segunda técnica o banco de filtros na escala Bark é deslocado 1 bark para cima para vozes femininas ou infantis, devido à constatação de que existe aproximadamente um bark de diferença na posição dos formantes das vozes masculinas e femininas. Na escala Mel, a posição dos filtros e a largura de suas bandas são adaptadas para acompanhar, também, a diferença na posição dos formantes. Somente após o deslocamento/adaptação dos filtros na escala Bark ou Mel, a análise espectral é realizada.

Em nosso caso, a primeira idéia para adaptar o sistema às características espectrais da voz do locutor, foi dividir o conjunto de locutores em vozes masculinas e femininas. Iria se treinar uma rede com vozes masculinas, outra com vozes femininas, e o sistema chavearia para um ou outro sistema em função do locutor ser homem ou mulher.

Para implementar esta idéia, o primeiro passo foi determinar a posição dos formantes de alguns sons da língua portuguesa utilizando as vozes masculinas e femininas da base de dados. A posição dos formantes foi determinada sobre os seguintes sons: **i** tônica, **a** tônica e **o** tônica. A razão de escolher estes sons foi a seguinte: **a**, **i** e **u** formam o *triângulo de vogais* (gráfico da freqüência do segundo formante em função da freqüência do primeiro formante), estando amplamente separadas no plano F_1-F_2. Isto minimiza a confusão perceptual destes sons (O'Shaughnessy, 1987). Como o som **o** na posição de sílaba tônica está mais presente em nosso corpus que o som **u**, o **o** foi escolhido em substituição do **u**.

As palavras escolhidas para determinar o valor dos formantes foram: **siga** para a **i** tônica, **rápido** para a **a** tônica e **mova-se** para a **o** tônica. Estas palavras pertencem à base de dados. A tabela seguinte, 5.1, mostra os valores médios dos três primeiros formantes na *região vocálica tônica* (RVT) destas palavras, para as vozes masculinas (m) e femininas (f) da base de dados.

Tabela 5.1: Freqüências médias dos três primeiros formantes (F_1, F_2 e F_3) em Hz, das regiões vocálicas tônicas das palavras **siga**, **rápido** e **mova-se** (vogais tônicas **i**, **a** e **o** respectivamente). Vozes masculinas (**m**) e femininas (**f**) da base de dados.

Palavra / vogal tônica	F_1 (Hz) ; m / f	F_2 (Hz) ; m / f	F_3 (Hz) ; m / f
siga / i	233 / 267	2085 / 2630	2800 / 3200
rápido / a	715 / 970	1270 / 1510	2320 / 2550
mova-se / o	380 / 450	780 / 890	2400 / 2790

Dado que a tabela acima corresponde aos valores médios dos formantes, os valores individuais, locutor a locutor, podem ter variações maiores ou menores que as indicadas. Portanto, se as diferenças em freqüência nos valores dos formantes da RVT não são significativas entre vozes masculinas e femininas, não é aconselhável utilizá-las como parâmetro discriminativo do sexo do falante. Com esta restrição, pode ser visto que para a **i** tônica, o deslocamento de freqüências é significativo no segundo e terceiro formantes, com valor entre 400 a 600 Hz. Para a **a** tônica, o deslocamento de freqüências é significativo nos três formantes, sendo de aproximadamente 200 Hz. Finalmente, para a **o** tônica o deslocamento de freqüência é significativo apenas no terceiro formante, sendo maior que 300 Hz.

Na escala mel (ver Tabela 3.1 do Cap. 3), estas diferenças implicam em:

a tônica:

1° formante masculino cai no 7° filtro mel, 1° formante feminino cai no 10° filtro mel

2° formante masculino cai no 12° filtro mel, 2° formante feminino cai no 13° filtro mel

3° formante masculino cai no 16° filtro mel, 3° formante feminino cai no 17° filtro mel

i tônica:

1° formante masculino cai no 2° filtro mel, 1° formante feminino cai no 3° filtro mel

2° formante masculino cai no 15° filtro mel, 2° formante feminino cai no 17° filtro mel

3° formante masculino cai no 17° filtro mel, 3° formante feminino cai no 18° filtro mel

o tônica:

1° formante masculino e feminino caem no 4° filtro mel

2° formante masculino cai no 8° filtro mel, 2° formante feminino cai no 9° filtro mel

3° formante masculino cai no 16° filtro mel, 3° formante feminino cai no 17° filtro mel

Em função destes resultados, decidimos utilizar como parâmetro discriminante das vozes masculinas e femininas o valor dos formantes da RVT da palavra **rápido**. Assim, para automa-

tizar o chaveamento do sistema para a rede masculina ou feminina, a primeira palavra que deveria ser falada é **rápido**. Seria determinada, então, a RVT desta palavra e se calculariam as posições dos formantes desta região. Antes de implementar um algoritmo que permitisse automatizar o chaveamento, foram feitos testes manuais para determinar a vantagem ou não de separar os locutores em masculinos e femininos. Os resultados obtidos são apresentados na próxima seção.

5.3 Separação dos locutores em masculinos e femininos

Nesta seção apresentar-se-ão os resultados obtidos utilizando-se como sistema de reconhecimento duas redes neurais, uma treinada com vozes masculinas e outra treinada com vozes femininas. O chaveamento para uma ou outra rede seria feito em função da *posição dos formantes* da região vocálica tônica de uma palavra chave. Esta palavra seria pronunciada no início pelo usuário do sistema. A palavra a utilizar seria **rápido**, por ter se constatado que na sua RVT os três primeiros formantes têm uma diferença de pelo menos 200 Hz entre as vozes masculinas e as femininas. Como mencionado, antes de implementar o chaveamento automático foram feitos testes manuais para determinar a vantagem ou não desta separação em locutores masculinos e femininos. Um inconveniente grave que teria a aplicação deste método, seria a necessidade de se dispor de um algoritmo eficiente para determinar a posição dos formantes, o que é não é tarefa fácil.

As redes neurais utilizadas nos testes foram os perceptrons multicamadas com uma camada escondida de 70 neurônios. O número de entradas da rede foi de 600 e o número de saídas 32. Para manter constante o número de entradas da rede neural utilizou-se a técnica de dizimação e interpolação de quadros apresentada no Cap. 4, seção 4.5. A análise foi realizada com quadros de 20 ms a cada 10 ms e janelas de Hanning. Na dizimação/interpolação de quadros utilizaram-se os seguintes parâmetros: número final de quadros: 50, fator de ponderação a utilizar no cálculo do delta-total: adev (desvio médio ou *average deviation*), número de quadros consecutivos a eliminar ou duplicar: ilimitado. Como parâmetros de entrada à rede neural utilizaram-se os mel-cepstrais calculados com o algoritmo de Davis e Mermelstein.

A razão de se escolher este tipo de análise, configuração de rede e parâmetros, foi por ter-se obtido com eles praticamente os melhores resultados do Cap. 4.

O sistema foi treinado com 16 locutores (8 masculinos e 8 femininos), e testado com 8 locutores (4 masculinos e 4 femininos), que não tinham participado do treinamento. Utilizou-se o vocabulário completo (dígitos mais comandos de cálculo mais comandos de movimento), e se obtiveram os seguintes resultados (taxas de erro E em valores percentuais):

1. Rede única treinada com vozes masculinas e femininas, testada com ambas vozes (E_{mf}), com vozes apenas masculinas (E_m) e com vozes apenas femininas (E_f):

   E_{mf} = 4.43% (Tabela 4.43, correspondente a análise com quadros de 20 ms a cada 10 ms)
   E_m = 3,39%
   E_f = 5,47%

   Pode ser visto que o sistema acerta mais para os homens que para as mulheres. Provavelmente isto é devido à faixa de freqüências empregada (3,4 kHz). Como as vozes femininas têm componentes de freqüência importantes na região do espectro acima de 3,4 kHz, esta informação discriminante é perdida.

2. Criando duas redes, uma para vozes masculinas e outra para vozes femininas (clusterização manual). Teste das duas redes tanto com vozes masculinas como com femininas:

   Rede masculina:

   E_m = 3,13%; queda em relação ao erro E_m = 3,39% da rede única: 7,67%.
   E_f = 24,74%

   Rede feminina:

   E_f = 4,95%; queda em relação ao erro E_f = 5,47% da rede única: 9,51%.
   E_m = 26,04%

   A taxa de erro global do sistema, supondo chaveamento correto para uma ou outra rede, ficou em 4,04%.

Como pode ser visto, houve um ganho em relação a ter uma rede única que incluísse todos os locutores. No caso da rede masculina, a queda na taxa de erro chegou a 7,67%, enquanto que na rede feminina esta queda chegou a 9,51%. Como contrapartida, as taxas de erro das re-

des quanto utilizadas com vozes pertencentes ao outro sexo aumentaram consideravelmente. Isto implica que o algoritmo utilizado para chavear para uma ou outra rede deve ser robusto e confiável, já que o chaveamento para a rede equivocada causará aumentos significativos na taxa de erro do sistema. Este algoritmo não foi implementado. A queda na taxa de erro global foi de 8,8% (de 4,43% no sistema com rede única, a 4,04% no sistema proposto).

Neste sistema ocorreu o mesmo que no caso da rede única: obtiveram-se maiores taxas de acerto com vozes masculinas que com vozes femininas.

Na próxima seção apresentar-se-ão os resultados obtidos ao separar os locutores em grupos espectralmente similares ao invés de em vozes masculinas e femininas.

5.4 Separação dos locutores em grupos espectralmente similares – Algoritmo de clusterização proposto

Estendendo a idéia anterior, decidiu-se não mais chavear para uma rede masculina ou feminina, mas para duas (ou mais) redes que agrupassem locutores espectralmente similares. Para determinar a semelhança espectral entre as vozes dos locutores, decidiu-se comparar um *vetor média* de parâmetros mel-cepstrais, obtido no centro da região vocálica tônica da palavra **rápido**. Neste caso implementou-se um algoritmo que fez a separação dos locutores automaticamente.

O algoritmo ideado e utilizado para a determinação da RVT e para o cálculo do vetor média de parâmetros mel-cepstrais foi o seguinte:

1. Cálculo das curvas de energia e delta-energia ao longo da palavra. Para isto utilizaram-se quadros de 20 ms a cada 10 ms. Nestes mesmos quadros calcularam-se os parâmetros mel-cepstrais utilizados depois como entradas da rede neural. Os quadros foram suavizados com janelas de Hanning.
2. Determinação da RVT utilizando as curvas de energia e delta-energia. Na RVT a curva de energia passa por um máximo, sendo que no início da RVT a curva de delta-energia tem um pico positivo e no final da RVT o delta energia tem um pico negativo (esta característica foi verificada utilizando-se as diferentes elocuções disponíveis da palavra rápido).

3. Delimitada a RVT, determinaram-se os 5 quadros centrais não consecutivos desta região e calculou-se o *vetor média* dos vetores de mel-cepstrais pertencentes a estes 5 quadros. Empregaram-se três elocuções da palavra **rápido** por cada locutor. Para cada elocução foi calculado o *vetor média* dos vetores de mel-cepstrais. Finalmente calculou-se a média dos *vetores média* das três elocuções. O vetor resultante foi utilizado para a clusterização do locutor respectivo na fase de reconhecimento. Na fase de treinamento utilizaram-se os três *vetores média* de cada locutor de treinamento (um por elocução), para agrupar os locutores em dois clusters.

Para agrupar os locutores foi utilizada uma rede de Kohonen com 12 entradas (tamanho do vetor de parâmetros mel-cepstrais) e dois neurônios de saída (quantizador vetorial com dois vetores). O algoritmo de treinamento utilizado foi o LVQ3. Após a clusterização nos dois grupos mencionados, os perceptrons multicamadas foram treinados com as vozes dos clusters respectivos.

Utilizando 16 locutores no treinamento (8 masculinos e 8 femininos) e 8 nos testes (4 masculinos e 4 femininos, que não participaram do treinamento), os resultados obtidos foram os seguintes:

1. Resultado da clusterização automática:

   a) No treinamento:

      Cluster 1: 9 locutores (7 masculinos, 2 femininos)

      Cluster 2: 7 locutores (6 femininos, 1 masculino)

   b) No teste:

      Cluster 1: 4 locutores (4 masculinos)

      Cluster 2: 4 locutores (4 femininos)

2. Taxas de erro E em % das redes 1 e 2:

   Rede 1: E_{11} (taxa de erro com locutores chaveados para a rede 1) = 2,87%

   E_{12} (taxa de erro com locutores chaveados para a rede 2) = 25,26%

   Rede 2: E_{22} (taxa de erro com locutores chaveados para a rede 2) = 4,43%

   E_{21} (taxa de erro com locutores chaveados para a rede 1) = 26,82%

   A taxa de erro global do sistema ficou em 3,65%.

A tabela seguinte, 5.2, mostra as quedas nas taxas de erro conseguidas com este sistema. São comparadas as taxas de erro do sistema atual, com as do sistema original (rede única) e com as do sistema que separa em locutores masculinos e femininos.

Tabela 5.2: Queda percentual nas taxas de erro, ao dividir os locutores em dois grupos espectralmente similares. A comparação é feita com as taxas de erro do sistema original (uma rede incluindo todos os locutores), e com as do sistema que separa em locutores masculinos e femininos.

	Queda em relação ao sistema original		Queda em relação ao sistema masc./ fem.	
Erros neste sistema	$E_m = 3,39\%$	$E_f = 5,47\%$	$E_m = 3,13\%$	$E_f = 4,95\%$
$E_{11} = 2,87\%$	15,34	—	8,31%	—
$E_{22} = 4,43\%$	—	~19%	—	10,51%

Pode ser visto que o sistema que agrupa locutores com vozes espectralmente similares, tem um desempenho bem melhor que os dos sistemas anteriores. Para a rede que contém maioria de locutores masculinos (cluster 1), a taxa de erro caiu 15,3% em relação ao sistema original e 8,3% em relação ao sistema que chaveia para rede masculina/feminina. Para a rede que contém maioria de locutores femininos (cluster 2), a taxa de erro caiu aproximadamente 19% em relação ao sistema original e 10,5% em relação ao sistema que chaveia para rede masculina/feminina.

Se consideramos a taxa de erro global do sistema, esta caiu 17,6% em relação ao sistema com rede única (de 4,43% a 3,65%), e 9,7% em relação ao sistema que separa em vozes masculinas e femininas (de 4,04% a 3,65%). Estas quedas nas taxas de erro são significativas.

Os melhores resultados que tínhamos obtido até agora correspondiam à análise síncrona com o pitch (Cap. 4, Seção 4.5.2). Naquela seção foi visto que utilizando dizimação e interpolação de quadros e análise síncrona com o pitch, a taxa de erro obtida era de 4,30% (Tabela 4.50). Este valor correspondia ao uso de parâmetros mel-cepstrais, uma rede backpropagation com 70 neurônios na camada escondida, número final de quadros igual a 50, número de quadros consecutivos a eliminar/duplicar ilimitado e uso de adev ou range como fator de ponderação no cálculo do delta-total.

Se comparamos essa taxa de erro de 4,30% com os valores obtidos aqui (4,04% para o sistema com rede masculina/feminina e 3,65% para o sistema que separa em dois clusters com vozes espectralmente similares), percebe-se a vantagem de utilizar adaptação ao locutor. As quedas nas taxas de erro são de aproximadamente 6,1% e 15,1% respectivamente.

Um resumo dos principais resultados obtidos é apresentado a seguir:

- **Erro utilizando quadros de análise de 20 ms a cada 10 ms = 4,43%**
- **Erro utilizando análise síncrona com o pitch = 4,30%**
- **Erro separando os locutores em vozes masculinas e femininas = 4,04%**
- **Erro separando os locutores em vozes espectralmente similares = 3,65%**

Como explicado no começo da seção, aqui foi utilizado chaveamento automático para uma ou outra rede. Para o chaveamento utilizou-se a palavra **rápido**, devendo ser esta a primeira palavra falada por qualquer usuário do sistema. O algoritmo de chaveamento mostrou-se efetivo.

Dado que o agrupamento dos locutores em dois clusters produziu taxas de acerto maiores que as do sistema original, numa segunda etapa testou-se o agrupamento em três clusters. O princípio utilizado para este agrupamento foi o mesmo utilizado anteriormente. Neste caso, porém, empregou-se uma rede de Kohonen com 12 entradas e 3 neurônios de saída (quantizador vetorial com 3 vetores). O algoritmo de treinamento foi o LVQ3.

Utilizando 16 locutores no treinamento (8 masculinos e 8 femininos) e 8 nos testes (4 masculinos e 4 femininos, que não participaram do treinamento), os resultados obtidos foram os seguintes:

1. Resultado da clusterização automática:

   a) No treinamento:

      Cluster 1: 7 locutores (6 masculinos, 1 feminino)

      Cluster 2: 4 locutores (2 masculinos, 2 femininos)

      Cluster 3: 5 locutores (5 femininos)

   b) No teste:

      Cluster 1: 3 locutores (3 masculinos)

      Cluster 2: 2 locutores (1 masculino, 1 feminino)

      Cluster 3: 3 locutores (3 femininos)

2. Taxas de erro E em % das redes 1, 2 e 3:

   Rede 1: E_{11} (taxa de erro com locutores chaveados para a rede 1) = 2,78%

   Rede 2: E_{22} (taxa de erro com locutores chaveados para a rede 2) = 4,69%

   Rede 3: E_{33} (taxa de erro com locutores chaveados para a rede 3) = 4,17%

   A taxa de erro global do sistema ficou em 3,78%.

Pode ser visto que, neste caso, o erro global foi um pouco maior que quando se agrupou em dois clusters (passou de 3,65% a 3,78%). A diferença é desprezível, mas a complexidade computacional aumenta. Desta forma, a clusterização em três grupos de locutores não é necessária.

Deve-se chamar a atenção aqui ao fato de que o número de locutores utilizados no treinamento e, portanto, na clusterização, foi pequeno (16 locutores, 8 homens e 8 mulheres). Assim, os resultados obtidos não podem ser generalizados para qualquer situação. Provavelmente, utilizando um número bem maior de locutores no treinamento, a clusterização em três ou mais grupos de locutores resulte em taxas de erro menores que as obtidas com 2 grupos. Quanto maior for o número de locutores, maior será a probabilidade de que existam mais de dois grupos de vozes espectralmente similares. Agrupando estas vozes em clusters independentes, a taxa de acerto do sistema deve aumentar.

Por outro lado, quanto maior for o número de clusters mais preciso e robusto deverá ser o algoritmo que chaveia o locutor para uma ou outra rede, já que o chaveamento para a rede errada fará com que a taxa de erro aumente consideravelmente. Pode ser visto nos resultados obtidos que quando o sistema é treinado com um grupo específico de vozes e é utilizado com outras vozes, a taxa de erro aumenta consideravelmente. Este fato nos abre uma nova possibilidade: alimentar todas as redes neurais com a palavra de chegada e levar em conta o resultado apenas da rede com maior índice de saída. A hipótese aqui é que nas redes que não correspondem ao locutor em questão, a saída ganhadora terá valor menor que o da rede correta. Neste caso, deve-se verificar não apenas a saída ganhadora de cada rede, como também seu valor para compará-los entre si. A vantagem residiria no fato de não ser mais necessário que o usuário do sistema fale inicialmente determinada palavra (aquela que o chaveia para a rede espectralmente mais parecida com a sua voz), podendo usar o sistema desde o início.

Esta hipótese não foi verificada e fica em aberto para futuras pesquisas.

Capítulo 6

Conclusões

O objetivo deste trabalho foi estudar e definir um sistema de reconhecimento de fala para palavras isoladas, independente do locutor e baseado em redes neurais artificiais. Para isto, vários aspectos foram pesquisados:

- que parâmetros utilizar como entradas da rede neural (algoritmos de obtenção dos parâmetros, complexidade computacional, índices de acerto obtidos, etc.);
- tipos de redes a utilizar (perceptrons multicamadas, redes de Kohonen-Grossberg, redes LVQ); número de camadas, número de neurônios, algoritmos de treinamento, etc.;
- tipos de análise do sinal de fala: análise com igual número de quadros para todas as palavras, análise com quadros de comprimento fixo, uso de quantização vetorial, *trace-segmentation*, etc.;
- dizimação e interpolação de quadros para manter constante o número de entradas da rede neural, critérios; análise com quadros de comprimento fixo e análise síncrona com o pitch;
- testes dos sistemas com e sem ruído;
- adaptação do sistema de reconhecimento de fala às características espectrais da voz do locutor.

1. No item ***parâmetros a utilizar como entradas da rede neural***, os parâmetros pesquisados foram: LPC, de reflexão, cepstrais, mel-cepstrais, energia, sonoridade, delta-energia, delta-cepstrais e delta-mel-cepstrais. Testou-se sua utilização em forma isolada e combinados entre si, visando obter a menor taxa de erro possível no sistema de reconhecimento. Para os parâmetros cepstrais e mel-cepstrais, foram testadas também, diferentes formas de obtê-los. Neste caso avaliou-se tanto a complexidade computacional dos diferentes algoritmos, como

o tempo de treinamento da rede neural e os índices de acerto obtidos. As conclusões a que se chegaram foram as seguintes:

Dentre todos os parâmetros testados, os mel-cepstrais foram os que permitiram obter taxas de acerto maiores, seguidos pelos cepstrais. A combinação de mel-cepstrais com energia permitiu, na maior parte dos casos, obter taxas de erro comparáveis ou menores às obtidas com os mel-cepstrais. O uso dos outros parâmetros piorou o desempenho do sistema.

No caso dos parâmetros delta (aproximações das derivadas dos parâmetros originais), seu uso não acrescentou poder discriminante à rede. Isto se justifica considerando que a informação que estes parâmetros carregam já está presente na rede –mesmo sem usá-los– como combinação linear dos parâmetros originais (isto ao realizar-se reconhecimento *estático* da palavra falada). Assim sua inclusão não incorpora informações novas à rede.

Quanto à *sonoridade*, seu uso permitiu, em alguns casos, melhorar os índices de reconhecimento dos parâmetros LPC e de reflexão, mas sem que com isto se superassem os índices de reconhecimento conseguidos com os mel-cepstrais. A combinação de sonoridade com parâmetros cepstrais ou mel-cepstrais prejudicou a capacidade discriminante da rede. Provavelmente isto é devido a erros na decisão sonoro/não-sonoro e ao fato de coexistir em alguns dos quadros tanto trechos sonoros como não-sonoros (em diversas elocuções de uma mesma palavra, isto provoca diferentes decisões quanto à sonoridade ou não de um quadro).

Também foi testado o uso de *subtração da média espectral*, já que em sistemas baseados em HMMs esta técnica tinha permitido melhorar os índices de acerto em até 2%. Testada aqui com os parâmetros mel-cepstrais, não produziu melhoras no desempenho do sistema.

Para o cálculo dos parâmetros mel-cepstrais, três algoritmos foram comparados: o que utiliza DCT (Davis & Mermelstein, 1980), o que utiliza FFT inversa (Deller et al., 1995), e o que utiliza a transformada bilinear. Dos três algoritmos, o de Davis & Mermelstein foi o que apresentou os melhores resultados, tanto no que diz respeito aos índices de reconhecimento, como ao tempo de treinamento do sistema (velocidade de convergência da rede neural). Quanto ao tempo de cálculo dos parâmetros, os obtidos através da transformação bilinear foram os mais rápidos de três (inclui-se aqui o tempo gasto para calcular os cepstrais via LPC, a partir dos quais são obtidos os mel-cepstrais). Porém, os índices de reconhecimento obtidos com eles foram os mais baixos.

Para o cálculo dos parâmetros cepstrais, dois algoritmos foram testados: o que os calcula via LPC e o que utiliza FFT. Os resultados foram praticamente os mesmos com leve vantagem para os obtidos via LPC. Quanto ao tempo de cálculo (complexidade computacional), os calculados via LPC foram bem mais rápidos de obter.

Em vista de tudo isto conclui-se que se o tempo de cálculo dos parâmetros não é um fator decisivo na escolha dos mesmos, então a melhor opção são os mel-cepstrais calculados com o algoritmo de Davis & Mermelstein. Se, pelo contrário, o tempo de cálculo é um fator limitativo, então é aconselhável usar os parâmetros cepstrais calculados via LPC.

Deve-se destacar aqui, que o pré-processamento do sinal de fala, que inclui a pré-ênfase, janelamento e cálculo dos parâmetros de entrada à rede neural, é responsável por mais de 95% do tempo total gasto em reconhecer a palavra (isto, utilizando perceptrons multicamadas). Portanto, o pré-processamento é o fator principal de demora, e o que consome mais recursos computacionais.

2. No item ***tipos de redes a utilizar*** foram pesquisados os perceptrons multicamadas, as redes de Kohonen, Kohonen-Grossberg e as LVQ. Para os perceptrons multicamadas foi pesquisado o uso de uma ou duas camadas escondidas, o número de neurônios delas, a função de ativação, e os valores da taxa de aprendizado e do fator momento. Para as redes de Kohonen e Kohonen-Grossberg, foi pesquisado o tipo de mapa a utilizar (linear ou quadrado), o número de neurônios dele, o raio de vizinhança e o fator de aprendizado. Para as redes LVQ pesquisou-se o valor da janela e a inicialização dos vetores (aleatórios mais Kohonen, ou vetores do conjunto de treinamento).

   Os melhores resultados se obtiveram com os perceptrons multicamadas treinados com o algoritmo backpropagation. As redes de Kohonen, Kohonen-Grossberg e LVQ tiveram desempenhos inferiores e foram mais sensíveis ao ruído que os perceptrons.

   Quanto ao número de camadas a utilizar nos perceptrons multicamadas, o uso de uma camada escondida produziu os melhores resultados. Com duas camadas escondidas a rede se tornou instável e os desempenhos foram inferiores. Em relação à função de ativação, o uso da sigmóide binária e da sigmóide bipolar produziu praticamente os mesmos resultados. A sigmóide binária foi utilizada na maior parte das simulações.

3. A respeito do ***tipo de análise a realizar no sinal de fala***, vários tipos foram testados: análise com igual número de quadros para todas as palavras, análise com quadros de comprimento fixo, uso de quantização vetorial, e uso de *trace-segmentation* e *individual trace-segmentation.*

   A análise com igual número de quadros para todas as palavras foi a que produziu os melhores resultados nesta etapa do trabalho. Quanto ao número de quadros a utilizar, o uso de 40 quadros/palavra permitiu obter as melhores taxas de acerto na maior parte dos casos. A análise com quadros de comprimento fixo teve desempenho quase sempre inferior e uma complexidade computacional maior (devido a que, usualmente, resulta em mais quadros de análise e portanto em mais parâmetros a calcular).

   Quanto à técnica de quantização vetorial, seu emprego não melhorou o desempenho do sistema. O mesmo pode se dizer da utilização das técnicas *trace segmentation* e *individual trace segmentation.*

4. Assim chegamos ao item ***dizimação e interpolação de quadros para manter constante o número de entradas da rede neural.***

   Um dos principais problemas abordados nesta tese, foi o das redes neurais terem um número fixo de entradas enquanto as palavras a reconhecer terem durações diferentes. Embora a utilização de um número fixo de quadros de análise seja uma solução possível a este problema, o objetivo era propor um método novo que permitisse obter taxas de acerto maiores. O método de dizimação e interpolação de quadros foi o caminho escolhido para este fim. Foi proposto aqui eliminar ou incorporar quadros, baseando-se na estabilidade espectral dos mesmos. Isto porque eliminar alguns quadros de uma região espectralmente estável (um som sonoro, por exemplo), não afeta significativamente a informação discriminante. Têm mais peso para o reconhecimento as descontinuidades espectrais (*transições* dos sons), que a *duração* de uma região estável. O mesmo pode se dizer no caso da interpolação. Interpolar significa incorporar quadros inexistentes a uma palavra. Com o critério utilizado, a incorporação de quadros é implementada duplicando-se quadros de uma região espectralmente estável, uma vez que a informação incorporada diz respeito, apenas, à duração desta região.

   Como critério de estabilidade escolheu-se a soma ponderada dos parâmetros delta-energia e delta-mel-cepstrais. A razão de se escolher estes parâmetros foi a verificação de que com am-

bos coeficientes é possível identificar a maior parte das transições que ocorrem na fala. Como fator de ponderação para o cálculo do delta-total foram testados três valores: o que iguala as variâncias de delta-energia e dos delta-mel-cepstrais, o que iguala seus desvios médios e o que iguala seus intervalos de variação. O resultado das simulações mostrou que os dois últimos fatores produzem resultados similares e melhores que os obtidos utilizando-se a variância.

Outros fatores pesquisados foram o número final de quadros a obter após a dizimação/interpolação e número máximo de quadros consecutivos a eliminar ou duplicar. Os melhores resultados se obtiveram com 50 quadros como objetivo da dizimação/interpolação, e não colocando limites ao número consecutivo de quadros a eliminar/duplicar. Com esta técnica se conseguiram resultados melhores que os obtidos até então, fazendo cair as taxas de erro em até 50% (19% para os comandos de cálculo, 43% para os de movimento, 50% para os dígitos e 32% para o vocabulário completo).

Até aqui, e ao longo da tese, a análise realizada no sinal de fala utilizava igual número de quadros em todas as palavras ou quadros de comprimento fixo. Este tipo de análise, como explicado no Cap. 5, apresenta dois problemas: o primeiro está relacionado à duração dos quadros de análise e o segundo à posição em que os quadros de análise caem dentro da palavra analisada. O primeiro problema afeta a resolução temporal e em freqüência da análise do sinal de fala. Uma boa resolução temporal exige quadros curtos, enquanto que uma boa resolução em freqüência requer quadros longos. Com a análise até aqui realizada estas condições nem sempre são alcançadas. O segundo problema diz respeito à distorção que pode ser introduzida devido aos quadros de análise caírem em qualquer posição dentro do sinal. Como os quadros são ponderados por janelas (Hanning, em nosso caso), se a região de decaimento da janela cai em cima de um pico de energia, o pico resultará atenuado. Da mesma forma uma região onde o sinal esteja caindo pode resultar em uma região estável, caso seja ponderada pela região de crescimento da janela. Assim, o cálculo dos parâmetros espectrais e temporais resulta dependente da posição da janela de análise.

Propôs-se, então, a análise *síncrona* com o pitch, que minimiza os dois problemas. Esta análise consiste em centrar as janelas nas marcas de pitch, fazendo que comecem na marca anterior e acabem na posterior. Nos trechos onde o sinal não é periódico colocam-se arbitrariamente marcas de pitch a cada 10 ms e procede-se como nas regiões periódicas. Desta forma a análise fica

casada com as características do sinal de fala: primeiro, a duração das janelas abrange sempre dois períodos de pitch, independentemente do valor deste. Isto permite obter um bom compromisso entre a resolução temporal e em freqüência da análise. Segundo, a posição das janelas minimiza-a distorção do sinal para o cálculo dos parâmetros.

Esta forma de análise produziu os melhores resultados. Comparando-os com os obtidos sem se aplicar a técnica de dizimação e interpolação de quadros, as quedas nas taxas de erro foram de até 84% (29% para os comandos de cálculo, 57% para os de movimento, 84% para os dígitos e 34% para o vocabulário completo). Assim, os resultados obtidos com a dizimação e interpolação de quadros empregando o método aqui proposto foram muito bons, e mostram a potencialidade da técnica.

5. Finalmente foi estudada a ***adaptação do sistema de reconhecimento de fala às características espectrais da voz do locutor***. Vários métodos tinham sido propostos na literatura visando normalizar o espectro da voz do locutor para um espectro médio, de forma a conseguir em *sistemas independentes do locutor* taxas de acerto próximas às obtidas em *sistemas dependentes do locutor.*

   Dois métodos propostos com este fim eram *frequency warping* e *Bark/Mel scale warping*. Na primeira técnica deforma-se o eixo de freqüências com uma regra linear ou exponencial, de forma que as posições dos formantes de um locutor específico estejam próximas das posições dos formantes de um espectro médio. Na segunda técnica o banco de filtros na escala Bark ou Mel é adaptado de forma a acompanhar, também, as diferenças nas posições dos formantes entre o espectro do falante e o espectro médio. Somente após o deslocamento/adaptação dos filtros na escala Bark ou Mel, a análise espectral é realizada.

   Propôs-se aqui um novo método de adaptação às características espectrais da voz do locutor. O método baseia-se no cálculo, em trechos específicos da fala do locutor, de um *vetor médio* de parâmetros mel-cepstrais. Este vetor chaveia o locutor para uma rede treinada com vozes espectralmente similares à dele. Com este método conseguiu-se diminuir as taxas de erro em até 15% (taxas de erro obtidas utilizando dizimação/interpolação de quadros).

Um resumo dos principais resultados obtidos ao longo desta tese é apresentado a seguir.

Tabela 6.1: Taxas de erro em % para os 4 vocabulários, utilizando redes backpropagation com uma camada escondida de 30 neurônios (dígitos, cálculo e movimento), e 50 neurônios (vocabulário completo); análise com 40 quadros por palavra e com quadros de 30 ms recalculados a cada 20 ms. Origem: tabelas 4.4, 4.5, 4.11 e 4.12.

Vocabulário	40 quadros	30 ms
Dígitos (D)	1,88	2,19
Cálculo (C)	3,03	3,41
Movimento (M)	2,65	4,92
D+C+M	6,51	8,72

Tabela 6.2: Taxas de erro em % para os 4 vocabulários, utilizando dizimação e interpolação de quadros e análise com janelas de Hanning de 20 ms a cada 10 ms. Rede neural: backpropagation com uma camada escondida de 30 neurônios (dígitos, cálculo e movimento), e 50 e 70 neurônios (vocabulário completo). Número de quadros consecutivos a eliminar/duplicar: ilimitado. Parâmetros de entrada: mel-cepstrais. numQ = 50. Origem: tabelas 4.40 a 4.43.

Vocabulário	h = 30	h = 50	h = 70
Dígitos (D)	0,94	–	–
Cálculo (C)	2,14	–	–
Movimento (M)	1,52	–	–
D+C+M	–	5,99	4,43

Tabela 6.3: Taxas de erro em % para os 4 vocabulários, utilizando dizimação e interpolação de quadros e análise síncrona com o pitch. Rede neural: backpropagation com uma camada escondida de 30 neurônios (dígitos, cálculo e movimento), e 50 e 70 neurônios (vocabulário completo). Número de quadros consecutivos a eliminar/duplicar: ilimitado. Parâmetros de entrada: mel-cepstrais. numQ = 50. Origem: tabelas 4.47 a 4.50.

Vocabulário	h = 30	h = 50	h = 70
Dígitos (D)	0,31	–	–
Cálculo (C)	1,89	–	–
Movimento (M)	1,14	–	–
D+C+M	–	5,86	4,30

Taxa de erro em % utilizando *adaptação às características espectrais da voz do locutor* (clusterizando em dois grupos de locutores) = 3,65%.

Finalmente podemos resumir as contribuições deste trabalho da seguinte forma: a técnica de dizimação e interpolação de quadros proposta é superior, sob o ponto de vista de taxas de acerto da rede, a quaisquer das técnicas apresentadas antes. Tanto a técnica que utiliza quadros de comprimento fixo como a que faz análise síncrona com o pitch, permitem obter resultados melhores que os obtidos analisando todas palavras com igual número de quadros, ou com quadros de comprimento fixo e completando as entradas vazias da rede com zeros. As melhoras no desempenho do sistema de reconhecimento são sensíveis. O custo computacional destas melhoras é elevado. Devem-se calcular parâmetros delta energia e delta mel-cepstrais em todos os quadros, com eles computar-se o delta total do quadro utilizando a ponderação devida, depois devem-se dizimar ou interpolar qua-

dros até se chegar ao número desejado e então alimentar-se a rede neural com os parâmetros correspondentes. No caso da análise síncrona com o pitch, há ainda o custo adicional de colocar marcas de pitch no sinal de fala e centrar as janelas de análise nelas. Neste caso a necessidade de um bom detector de pitch é primordial para o sucesso da análise. Em ambos casos (análise síncrona com o pitch ou com quadros de comprimento fixo), o processamento só pode ser realizado após ter chegado a palavra completa, devido à necessidade de se definir quantos quadros devem ser eliminados ou duplicados. O ganho no resultado, porém, é significativo. Um aspecto que deve ser analisado mais profundamente neste item é a forma de determinar as regiões espectralmente estáveis e instáveis de uma palavra, já que nem sempre as transições (regiões a preservar), são colocadas em evidência pelos coeficientes delta.

Em relação à adaptação do sistema de reconhecimento de fala às características espectrais da voz do locutor, os resultados obtidos também são promissores. É provável que utilizando um número bem maior de locutores no treinamento, a clusterização em três ou mais grupos de locutores resulte em taxas de erro menores que as obtidas com 2 grupos. Quanto maior for o número de locutores, maior será a probabilidade de que existam mais de dois grupos de vozes espectralmente similares. Agrupando estas vozes em clusters independentes, a taxa de acerto do sistema deve aumentar. Por outro lado, quanto maior for o número de clusters mais preciso e robusto deverá ser o algoritmo que chaveia o locutor para uma ou outra rede, já que o chaveamento para a rede errada fará com que a taxa de erro aumente consideravelmente.

Neste item fica por pesquisar a possibilidade de alimentar todas as redes neurais com a palavra de chegada e levar em conta o resultado, apenas, da rede com maior índice de saída. A hipótese aqui é que nas redes que não correspondem ao locutor em questão, a saída ganhadora terá valor menor que o da rede correta. Neste caso, deve-se verificar não apenas a saída ganhadora de cada rede, como também seu valor para poder compará-los entre si. A vantagem residiria no fato de não ser mais necessário que o usuário do sistema fale inicialmente determinada palavra (aquela que o chaveia para a rede espectralmente mais parecida com a sua voz), podendo usar o sistema desde o início.

Referências

Bourlard, H and J. Wellekens (1990). "Links between Markov Models and Multilayer Perceptrons". *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 12(12):1167-1178.

Cabral, E. and G. Tattersall (1995). "Trace Segmentation of Isolated Utterances for Speech Recognition". *International Conference on Audio Speech and Signal Processing, Detroit Michigan, USA.* Vol. 1, pp. 365-368.

Charpentier, F and E. Moulines (1989). "Nouvelles Techniques de Synthèse de la Parole". *L'écho des Recherches,* N° 137, pp. 37-46.

Davis, S. and P. Mermelstein (1980). "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences". *IEEE Trans. on ASSP,* Vol. 28, N° 4, pp. 357-366.

Deller, J., J. Proakis and J. Hansen (1993). *Discrete-Time Processing of Speech Signals.* McMillan Publishing Co.

Eide, E. and H. Gish (1996). "A parametric approach to vocal tract length normalization". *Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Atlanta, Georgia, USA.* Vol. 1, pp. 346-348.

Fant, G. (1973). *"Speech sounds and features".* MIT Press.

Fausett, L. (1994). *Fundamentals of Neural Networks.* Prentice-Hall, Inc. Englewood Cliffs, New Jersey.

Furui, S. (1995). "Speech Recognition – Past, Present, and Future –". *NTT Review,* Vol. 7, N° 2, pp. 13-18.

Grossberg, S. (1969). "Some networks that can learn, remember and reproduce any number of complicated space-time patterns." *Journal of Mathematics and Mechanics,* 19: 53-91.

Hecht-Nielsen, R. (1987a). "Kolgomorov's Mapping Neural Network Existence Theorem." *IEEE First International Conference on Neural Networks, San Diego, CA, USA.* III: 11-14.

Hecht-Nielsen, R. (1987b). "Counterpropagation Networks." *Applied Optics,* 26(23): 4979-4984.

Hecht-Nielsen, R. (1987c). "Counterpropagation Networks." *IEEE First International Conference on Neural Networks, San Diego, CA, USA.* II:19-32.

Hecht-Nielsen, R. (1988). "Applications of Counterpropagation Networks." *Neural Networks*, 1(2): 131-139.

Hecht-Nielsen, R. (1990). *Neurocomputing*. Reading, MA: Addison-Wesley.

Junqua, J. C. and J. P. Haton (1996). *Robustness in Automatic Speech Recognition*. Kluwer Academic Publishers.

Kamm, T., G. Andreou and J. Cohen (1995). "Vocal tract normalization in speech recognition compensating for systematic speaker variability". *Proceedings of the 15th Annual Speech Research Symposium, CLSP, Johns Hopkins University, Baltimore, MD, USA*. pp. 175-178.

Katagiri, S. and C-H. Lee (1993). "A new Hybrid Algorithm for Speech Recognition based on HMM Segmentation and Learning Vector Quantization". *IEEE Transactions on Speech and Audio Processing*, 1(4):421-430.

Kohonen, T. (1989). *Self-Organization and Associative Memory* (3rd ed.), Berlin: Springer-Verlag.

Kohonen, T. (1990a). "The Self-Organizing Map." *Proceedings of the IEEE*, Vol. 78, N° 9, pp. 1464-1480.

Kohonen, T. (1990b). "Improved Versions of Learning Vector Quantization." *International Joint Conference on Neural Networks, San Diego, CA, USA*. I: 545-550.

Kurt Schäfer-Vincent (1983). "Pitch Period Detection and Chaining: Method and Evaluation". *Phonetica*, N° 40, pp. 177-202.

Lee, L. and R. Rose (1996). "Speaker normalization using efficient frequency warping procedures". *Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Atlanta, Georgia, USA*. Vol. 1, pp. 353-356.

Linde, J., A. Buzo and R. M. Gray (1980). "An Algorithm for Vector Quantizer", *IEEE Trans. On Communications*, Vol. 28, N° 1, pp. 84-94.

Martins J. A. (1997). *Avaliação de Diferentes Técnicas para Reconhecimento de Fala* (Tese de Doutorado). Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação. SP, Brasil.

McCulloch, W. S. and W. Pitts (1943). "A logical Calculus of the Ideas Immanent in Nervous Activity." *Bulletin of Mathematical Biophysics*, 5: 115-133. Reprinted in Anderson & Rosenfeld [1988], pp. 18-28.

Minsky, M. S. and S. Papert. (1988). *Perceptrons, Expanded Edition*. Cambridge, MA: MIT Press. Original Edition: 1969.

Morgan, N. and H. Bourlard (1990). "Continuous Speech Recognition using Multilayer Perceptrons with Hidden Markov Models". *Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Albuquerque, New Mexico, USA.* Vol. 1, pp. 413-416.

Morgan, N. and H. Bourlard (1995). "Neural Networks for Statistical Recognition of Continuous Speech". *Proceedings of the IEEE,* 83(5):25-42.

Nejat Ince, A., editor (1992). *Digital Signal Processing, Speech Coding, Synthesis and Recognition.* Kluwer Academic Publishers.

Niles, L.T. and H. F. Silverman (1990). "Combining Hidden Markov Model and Neural Network Classifiers". *Proceedings of the International Conference on Acoustics Speech and Signal Processing, Albuquerque, New Mexico, USA.* Vol. 1, pp. 417-420.

O'Shaughnessy, D. (1987). *Speech Communication: Human and Machine.* New York: Addison-Wesley.

Parker, D. (1985). *Learning Logic.* Technical Report TR-87, Cambridge, MA, USA: Center for Computational Research in Economics and Management Science, MIT.

Picone, J. (1993). "Signal Modeling Techniques in Speech Recognition". *Proceedings of the IEEE,* Vol. 81, Nº 9, pp. 1215-1247.

Rabiner, L. and R. Schafer (1978). *Digital Processing of Speech Signals.* Prentice Hall Inc.

Rabiner, L. (1989). "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition." *Proceedings of the IEEE,* Vol. 77, Nº 2, pp. 257-286.

Rabiner, L. and Juang B-H. (1993). *Fundamentals of Speech Recognition.* Prentice Hall, Englewood Cliffs, New Jersey.

Rosenblatt, F. (1958). "The Perceptron: a Probabilistic Model for Information Storage and Organization in the Brain". *Psychological Review,* 65: 386-408. Reprinted in Anderson & Rosenfeld [1988], pp. 92-114.

Rosenblatt, F. (1962). *Principles of Neurodynamics.* New York: Spartan.

Rumelhart, D. E., G. E. Hinton and R. J. Williams. (1986). "Learning Internal Representations by Error Propagation." In D. E. Rumelhart & J. L. McClelland, eds., *Parallel Distributed Processing,* Vol. 1, Chapter 8, reprinted in Anderson & Rosenfeld [1988], pp. 675-695.

Runstein, F. e F. Violaro (1993). "Reconhecimento de Dígitos Isolados Utilizando Redes Neurais". *Anais do X Congreso Chileno de Ingeniería Eléctrica,* Valdivia, Chile. pp. J41-J44.

Runstein, F. e F. Violaro (1995). "An Isolated-Word Speech Recognition System Using Neural Networks". *"Proceedings of the 38th Midwest Symposium on Circuits and Systems"*, Rio de Janeiro, Brasil, Vol. 1, pp. 550-553.

Runstein, F., F. Violaro e H. F. Nunes (1995). "Uso de Diferentes Parâmetros de Entrada em um Sistema de Reconhecimento de Fala Baseado em Redes Neurais". *Anais do 13° Simpósio Brasileiro de Telecomunicações*, Águas de Lindóia, São Paulo, Brasil. Vol. 1, pp. 155-160.

Runstein, F., F. Violaro and C. H. da Silva (1997). "A Speaker Independent Speech Recognition System Based on Neural Networks". *An International Journal on Latin American Applied Research*, Vol. 27, N° 3, pp. 139-142, e anais da *VII Reunión de Trabajo en Procesamiento de la Información y Control*, San Juan, Argentina, Setembro de 1997.

Shynk, J. J. and N. J. Bershad (1991). "Steady-state analysis of a single-layer perceptron, based on a system identification model with bias terms." *IEEE Transactions on Circuits and Systems*, CAS-38, pp. 1030-1042.

Shynk, J. J. and N. J. Bershad (1992). "Stationary points and performance surfaces of a perceptron learning algorithm for a nonstationary data model." *International Joint Conference on Neural Networks, Baltimore, MD, USA*. Vol. 2, pp. 133-139.

Werbos, P. (1974). *Beyond Regression: New Tools For Prediction and Analysis in the Behavioral Sciences* (Ph.D. Thesis). Cambridge, MA: Harvard U. Committee on Applied Mathematics.

Waibel A., T. Hanazawa, G. Hinton, K. Shikano and K. Lang (1989). "Phoneme Recognition Using Time-Delay Neural Networks." *IEEE Trans. on ASSP*, Vol. 37, N° 3, pp. 328-339.

Zhan P. and Alex Waibel (1997). "Vocal tract length normalization for large vocabulary continuous speech recognition". *Computer Science Technical Report : CMU-CS-97-148.* Carnegie Mellon University, Pittsburgh, PA 15213, USA.