

João Henrique Burckas Ribeiro

Desenvolvimento de uma Técnica de Reconhecimento de
Padrões Baseada em Distância

Florianópolis

2003

Universidade Federal de Santa Catarina

**Programa de Pós-Graduação
em Engenharia Elétrica**

**Desenvolvimento de uma Técnica de Reconhecimento de
Padrões Baseada em Distância**

Dissertação submetida à
Universidade Federal de Santa Catarina
como parte dos requisitos para a
obtenção do grau de Mestre em Engenharia Elétrica.

João Henrique Burckas Ribeiro

Florianópolis, maio de 2003

Desenvolvimento de uma Técnica de Reconhecimento de Padrões Baseada em Distância

João Henrique Burckas Ribeiro

‘Esta Dissertação foi julgada adequada para obtenção do Título de Mestre em Engenharia Elétrica, Área de Concentração em *Engenharia Biomédica*, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina.’

Fernando Mendes de Azevedo, D.Sc.
Orientador

Edson Roberto de Pieri, Dr.
Coordenador do Programa de Pós-Graduação em Engenharia Elétrica

Banca Examinadora:

Fernando Mendes de Azevedo, D.Sc.
Presidente

Jefferson Luiz Brum Marques, PhD.

Fernanda Isabel Marques Argoud, Dr. Eng.

Joceli Mayer, PhD.

A minha mãe, Hilda Burckas Ribeiro
e a meu pai, Manuel dos Santos Fernandes Ribeiro.

Resumo da Dissertação apresentada à UFSC como parte dos requisitos necessários para a obtenção do grau de Mestre em Engenharia Elétrica.

Desenvolvimento de uma Técnica de Reconhecimento de Padrões Baseada em Distância

João Henrique Burckas Ribeiro

Maio/2003

Orientador: Fernando Mendes de Azevedo, D.Sc.

Área de Concentração: Engenharia Biomédica

Palavras-chave: Reconhecimento de padrões, inteligência artificial, redes neurais

Número de páginas: 80

RESUMO: Técnicas de reconhecimento de padrões são importantes ferramentas dentro da inteligência artificial, sendo aplicáveis em áreas como análise de imagens, reconhecimento de caracteres, reconhecimento de fala, auxílio a diagnósticos médicos, identificação de pessoas, inspeção industrial.

Neste trabalho foram desenvolvidas quatro técnicas de reconhecimento de padrões baseadas no cálculo de distância, sendo três delas não paramétricas (PID, SID, MID) e a quarta técnica, a elipsoidal, paramétrica. Esta última técnica pode ser considerada a otimização das três primeiras.

Para verificar a validade das técnicas desenvolvidas fez-se um estudo das técnicas de reconhecimento de padrões. Aqui, são apresentadas as principais técnicas: regra de Bayes, máxima verossimilhança, aproximação bayesiana, vizinhança mais próxima (k-NN), Parzen window, perceptron multicamadas, redes RBF e mapas de Kohonen. Em seguida, algumas dessas técnicas foram comparadas com as desenvolvidas aqui.

Para fazer essa comparação, foi criado o software *classificador*, que mostrou ser uma ferramenta útil para o projeto de sistemas de reconhecimento de padrões, pois possibilita testar diferentes técnicas, verificando qual a técnica é mais adequada para cada problema.

Essa comparação mostra que as técnicas PID, SID, MID e elipsoidal têm bom desempenho e que podem ser alternativas a considerar-se em projetos de sistemas de reconhecimento de padrões.

Abstract of Dissertation presented to UFSC as a partial fulfillment of the requirements for the degree of Master in Electrical Engineering.

A Pattern Recognition Technique Based on Distance

João Henrique Burckas Ribeiro

March/2003

Advisor: Fernando Mendes de Azevedo, D.Sc.

Area of Concentration: Biomedical Engineering

Key-words: Pattern Recognition, Artificial Intelligence, Neural Networks

Page numbers: 80

Abstract: Pattern recognition techniques are important artificial intelligence tool. These techniques are applied on some areas as: image analysis, optical character recognition (OCR), speech recognition, diagnosis support, person identification, industrial inspection.

Four pattern recognition techniques based on distance were developed, three of them are nonparametric (PID, SID and MID) and the fourth technique, the elipsoidal, is parametric. The elipsoidal technique can be considered an optimization of PID, SID and MID.

The majors pattern recognition techniques were studied and presented here: Bayes rule, maximum likelihood, Bayesian approach, k-nearest neighbor (k-NN), Parzen window, multi-layer perceptron, RBF neuronal network and Kohonen map. After that, the developed techniques were compared with some of these traditional techniques.

The software Classificador was developed to compare these techniques and it showed to be an useful tool to pattern recognition systems design. The software Classificador enables to test different pattern recognition techniques, and verify which technique is more suitable to each particular problem.

This comparison showed that the techniques PID, SID, MID and elipsoidal have a good performance and can be possibilities to consider in pattern recognition systems design.

Sumário

Lista de Figuras.....	vi
Lista de Tabelas e Gráficos.....	viii
Notação.....	x
1 Introdução.....	1
1.1 Motivação.....	2
1.2 Objetivos.....	3
2 Fundamentação Teórica.....	4
2.1 Sistema de reconhecimento de padrões.....	5
2.2 Projeto de sistema de reconhecimento de padrões.....	7
2.3 Complexidade Computacional.....	11
3 Técnicas de Reconhecimento de Padrões.....	13
3.1 Regra de Bayes.....	13
3.2 Distribuição normal.....	15
3.3 Máxima Verossimilhança (Maximum Likelihood).....	16
3.4 Aproximação Bayesiana.....	18
3.5 Paramétrica X Não Paramétrica.....	19
3.6 Vizinhaça mais Próxima (Nearest Neighbor).....	20
3.7 Parzen Window.....	22
3.8 k-Médias.....	24
3.9 Redes Neurais.....	24
3.9.1 Modelo de Neurônio.....	25
3.9.2 Perceptrons de múltiplas camadas.....	26
3.9.3 Redes RBF.....	28
3.9.4 Rede Kohonen.....	29
4 Técnicas Propostas.....	32
4.1 Formalizando e refinando as técnicas propostas.....	34
4.2 PID (Produtório do Inverso das Distâncias).....	35
4.3 SID (Somatório do Inverso das Distâncias).....	36
4.4 MID (Mínima Distância).....	36
4.5 Elipsoidal.....	36
5 Softwares.....	44
5.1 Classificador.....	44
5.2 Mapeamento.....	50
6 Estudo Comparativo.....	53
7 Estudo de caso.....	76
8 Discussão.....	79
9 Conclusão.....	83
10 Futuros trabalhos.....	86
11 Bibliografia.....	87

Lista de Figuras

Figura 2.1 - Quatro caligrafias diferentes entre as diversas caligrafias possíveis para o alfabeto.....	4
Figura 2.2 - O desenhista reproduz as características básicas que permitam ao observador reconhecer os traços-padrão de uma garrafa ou de uma casa.	5
Figura 2.3 - Diagrama de blocos de um Sistema de Reconhecimento de Padrões.....	6
Figura 2.4 - Etapas principais de um sistema de reconhecimento de padrões.....	8
Figura 2.5 - Fronteiras de decisão: a) Muito simples, só resolve problemas lineares.....	9
Figura 3.1 - Diagrama de blocos dos classificadores estudados e os propostos. O significado das siglas PID, SID e MID são respectivamente: Produtório do Inverso das Distâncias, Somatório do Inverso das Distâncias e Mínima Distância.....	13
Figura 3.2 - Representação gráfica dos cálculos das diferentes distâncias.....	21
Figura 3.3 - Simulação Parzen window com diferentes valores de h.....	23
Figura 3.4 - Neurônio artificial.....	26
Figura 3.5 - Rede perceptron de múltiplas camadas.....	26
Figura 3.6 - Rede RBF.....	28
Figura 3.7 - Rede Kohonen (SOM).....	30
Figura 4.1 - Ilustração dos métodos PID, SID e MID.....	32
Figura 4.2 - Aglomeração circular.....	38
Figura 4.3 - Aglomeração elíptica.....	39
Figura 4.4 - Fluxograma do algoritmo de busca dos elipsóides.....	42
Figura 4.5 - Passos do fluxograma de busca dos elipsóides.....	42
Figura 5.1 - Exemplo do formato CSV.....	45
Figura 5.2 - Tela de entrada de dados do software classificador.....	46
Figura 5.3 - Tela para gerar amostras pseudo-aleatórias no software classificador.....	47
Figura 5.4 - Janela de escolha dos métodos de reconhecimento de padrões.....	48
Figura 5.5 - Treinamento de rede perceptron multi-camadas.....	49
Figura 5.6 - Palheta “Regra de Bayes”.....	49

Figura 5.7 - Palheta de seleção dos métodos PID, SID, MID e elipsoidal.....	50
Figura 5.8 - Gráfico de superfície = gráfico de cores.....	51
Figura 5.9 -Programa Mapeamento.....	52
Figura 6.1 - OU exclusivo primário.....	54
Figura 6.2 - OU exclusivo, Teste 3.....	55
Figura 6.3 - Mapas de resposta do teste 2.....	66
Figura 6.4 - Amostras do teste 10.....	69
Figura 6.5 - Mapas de resposta do teste 10.....	71
Figura 6.6 - Mapas de resposta do teste 11.....	73

Lista de Tabelas e Gráficos

Tabela 6.1 - OU exclusivo.....	53
Tabela 6.2 - Teste 1, realizado com amostras com baixo desvio-padrão e poucas amostras de exemplo. O número de amostras de teste foi 404 para todos os testes, independentemente do número de amostras de exemplo.....	57
Gráfico 6.1 - Percentual de acerto do teste 1 - Percentual de acerto de cada método. Possivelmente por causa do baixo número de amostras, os métodos MID e o elipsoidal não tiveram um desempenho tão bom.	57
Tabela 6.3 - Teste 2, realizado com amostras com baixo desvio-padrão e médio número de amostras de exemplo. O número de amostras de teste foi mantido em 404.....	58
Gráfico 6.2 - Percentual de acerto do teste 2. Com o aumento do número de amostras de teste, pode-se notar que, em média, o percentual de acerto aumentou. Nesse teste destaca-se o método elipsoidal que no teste 1 teve baixo desempenho, já aqui teve melhor desempenho.....	58
Tabela 6.4 - Teste 3, realizado com amostras com baixo desvio-padrão e um grande número de amostras de exemplo.	59
Gráfico 6.3 - Percentual de acerto do teste 3. O percentual de acerto nesse teste foi mais bem equalizado em relação aos testes anteriores.....	59
Tabela 6.5 - Teste 4, realizado com amostras com médio desvio-padrão e poucas amostras de exemplo.....	60
Gráfico 6.4 - Percentual de acerto do teste 4. Com o aumento do desvio-padrão, a média do percentual de acerto diminuiu. Neste caso, há um destaque para o método Parzen window que obteve o melhor desempenho.....	60
Tabela 6.6 - Teste 5, realizado com amostras com médio desvio-padrão e número de amostras de exemplo.....	61
Gráfico 6.5 - Percentual de acerto do teste 5, que teve em média um percentual maior que o teste 4 simplesmente pelo aumento do número de amostras de exemplo.....	61
Tabela 6.7 - Teste 6, realizado com amostras com médio desvio-padrão e muitas amostras de exemplo.....	62
Gráfico 6.6 - Percentual de acerto do teste 6. Com um número ainda maior de amostras a maioria dos métodos apresentou uma melhora no percentual de acerto, entretanto, foi difícil encontrar uma boa configuração para treinar a rede neural.....	62
Tabela 6.8 - Teste 7, realizado com amostras com alto desvio-padrão e poucas amostras de exemplo.	63
Gráfico 6.7 - Percentual de acerto do teste 7. Esse é o teste que obteve a menor média de percentual de acerto, em razão do alto desvio-padrão e baixo número de amostras.....	63

Tabela 6.9 - Teste 8, realizado com amostras com alto desvio-padrão e um número médio de amostras de exemplo.....	64
Gráfico 6.8 - Percentual de acerto do teste 8. Nesse teste o método pela curva normal obteve um resultado muito superior em relação aos outros.....	64
Tabela 6.10 - Teste 9, realizado com amostras com alto desvio-padrão e um número alto de amostras de exemplo.....	65
Gráfico 6.9 - Percentual de acerto do teste 9. Por causa de um grande número de amostras com o desvio-padrão elevado, foi muito complicado encontrar uma configuração para a rede neural que treinasse; em consequência, o percentual de acerto da rede caiu muito.....	65
Tabela 6.11 - Lógica "E".....	68
Tabela 6.12 - Teste 10, realizado principalmente para demonstrar a solução de problemas de reconhecimento de padrões pela curva normal, resolve apenas casos específicos.....	69
Gráfico 6.10 - Percentual de acerto do teste 10, cujo resultado foi muito diferente dos testes de 1 a 9; isso mostra que para cada tipo de problema de reconhecimento de padrões pode haver um tipo de solução diferente, não havendo uma solução boa para todos os tipos de problemas.....	70
Tabela 7.1 - Resultado do estudo de caso. Quando classificou-se como glicemia normal; quando , classificou-se como hipoglicemia.....	77
Gráfico 7.1 - Percentual de acerto da detecção da hipoglicemia.....	77
Tabela 9.1 - Resumo da comparação entre os métodos de reconhecimento de padrões.....	84

Notação

α	ângulo da equação sigmoïdal
φ	função janela
$\lambda(c_i c_j)$	função perda: prejuízo atribuído a uma classificação errônea, ou seja, quando se classifica o objeto como c_i enquanto o correto seria c_j
\mathfrak{R}	risco condicional
σ	desvio-padrão
σ^2	variância
Σ	matriz de covariância
θ	parâmetro de uma função
$\boldsymbol{\theta}$	vetor de parâmetros de uma função
μ	média de um escalar
$\boldsymbol{\mu}$	média de um vetor
$\nabla_{\boldsymbol{\theta}}$	operador gradiente
$\nabla_{\boldsymbol{\theta}} l$	gradiente do logaritmo da verossimilhança
c	classe
\mathbf{c}	vetor de classes
C	número de classes
\mathbf{c}_e	vetor do centro de uma função radial
c_{e_j}	uma dimensão do vetor \mathbf{c}_e
D	dimensão de entrada do classificador e, por conseqüência, dimensão do vetor \mathbf{x}
D_E	distância euclideana
D_{MH}	distância Manhattan
D_{ML}	distância Mahalanobis

E	elipsóide
FA, FB	focos do elipsóide
g	grau de generalização
h	aresta do hipercubo formado pela função janela pelo método Parzen window
k	número de amostras mais próximo nas equações da vizinhança mais próxima e Parzen window; pode também ser usado como índice em algumas equações
k_i	número de amostras da classe c_i entre as k amostras mais próximas
$l(\theta)$	logaritmo da verossimilhança
M	dimensão de saída do classificador e por consequência dimensão do vetor \mathbf{y}
N	número de amostras
N_c	número de amostras de uma única classe
$O(.)$	indica a ordem de uma função, usado para comparar a complexidade computacional entre os algoritmos
p	dimensão do vetor θ
$P(c)$	probabilidade <i>a priori</i> de ocorrer a classe c
$p(\mathbf{x}/c)$	densidade de probabilidade da classe c em função de \mathbf{x}
$P(c/\mathbf{x})$	probabilidade <i>a posteriori</i> : probabilidade de \mathbf{x} pertencer à classe c
Q	número de amostras cuja saída y correspondente tende a 1
R	número de amostras cuja saída y correspondente tende a 0
R_E	raio do elipsóide
\mathbf{s}	conjunto de amostras com todas as classes
\mathbf{s}_q	vetores de amostras apenas cuja saída y correspondente é 1
\mathbf{s}_r	vetores de amostras apenas cuja saída y correspondente é 0
\mathbf{S}	conjunto de amostras de uma única classe

S^n	número de amostras de uma única classe onde n indica o número de amostras
T_m	inverso das distância ponderadas referente à saída y_m
u	somatório das excitações dos neurônios
V	volume que envolve as k amostras capturadas nos métodos vizinhança mais próxima e Parzen window
w	peso das ligações entre neurônios
x	variável de entrada (característica)
\mathbf{x}	vetor de entrada (vetor de característica)
y	saída do classificador
\mathbf{y}	vetor de saída do classificador

1 Introdução

A todo momento estamos fazendo “reconhecimento de padrões” em nossas vidas. Neste exato instante, ao ler este texto, você reconhece estes traços pretos como letras, reconhece um grupo de letras como uma palavra e reconhece o significado de cada palavra. Enfim, reconhecer o rosto de uma pessoa, distinguir um cachorro de um gato, compreender a fala, ler as mais diversas caligrafias e até mesmo interpretar um exame de eletrocardiografia, tudo isso é reconhecimento de padrões.

Tornar máquinas capazes de reconhecer padrões possibilita automatizar todas essas tarefas. As técnicas de reconhecimento de padrões têm aplicação em diferentes áreas, mas os estudos são naturalmente focados nas aplicações e não nas técnicas de reconhecimento em si. Por isso, às vezes, não é fácil reunir o que se sabe sobre reconhecimento de padrões.

Na medicina, técnicas de reconhecimento de padrões aplicam-se, de maneira geral, apoiando e tornando mais ágil o trabalho do médico e dando-lhe maior segurança para fazer diagnóstico.

Um dos desafios para os engenheiros biomédicos é desenvolver sistemas para auxiliarem no diagnóstico médico de forma automática e confiável. Para isso ser possível utilizam-se técnicas de reconhecimento de padrões. O uso dessas técnicas aproxima o desempenho da máquina à capacidade humana. As máquinas são muito eficientes para filtrar, calcular e extrair parâmetros numéricos de sinais biomédicos, mas, para associar esses parâmetros a um diagnóstico, os médicos são muito mais eficientes. Em outras palavras, os médicos são capazes de reconhecer padrões em parâmetros biomédicos; já as máquinas em princípio, não. Técnicas de reconhecimento de padrão procuram tornar as máquinas quase tão capazes quanto os médicos; mas, é claro, as máquinas serão só auxiliares, já que não serão confiáveis para problemas que escapem aos padrões para os quais foram treinadas. Essas técnicas são constante alvo de estudos da inteligência artificial.

1.1 Motivação

O objetivo final que motiva o trabalho discutido nesse livro é um modelo bem balanceado de um sistema cognitivo pensante, inteligente e flexível. Tal modelo deve ser capaz de se comunicar em linguagem natural como o inglês, perceber e reconhecer objetos em seu redor e responder apropriadamente a eles, inferir sobre o que se vê, conhecer, lembrar, resolver problemas, enfim, aprender a conduzir-se de tal forma a maximizar as coisas de valor. Isso, caso queiramos modelar seres humanos ou construir computadores inteligentes.

Leonard Uhr em “Pattern Recognition, Learning and Thought” [UHR 1973].

Desenvolver uma inteligência artificial tão capaz quanto a nossa fascina cientistas há anos, não só pelas aplicações que proporcionaria mas, principalmente, pela possibilidade de compreender o funcionamento de nossa própria inteligência. Reconhecer padrões é só uma das capacidades da inteligência natural, mas, sem dúvida nenhuma, é uma das principais.

Muitas vezes, deseja-se que máquinas reconheçam padrões. Por exemplo, na medicina, sistemas com tal capacidade podem auxiliar no diagnóstico, no tratamento, no estudo da interação de drogas no organismo, no processamento de sinais neurobiológicos, etc [SILVA 1998][GARCIA 2001] [ARGOUD 2001][IAIONE 2002].

1.2 Objetivos

Ao processar sinais em engenharia biomédica, com frequência nos deparamos com problemas de reconhecimento de padrões. Para reconhecer cardiopatias pelo processamento de eletrocardiogramas (ECGs), características epilépticas em eletroencefalograma (EEG) ou, até mesmo, reconhecer possível tumor em ultrassonografia, entre tantas outras possibilidades [PATRICK 1972].

Este trabalho tem por objetivo: a) fornecer, previamente, breve introdução ao estudo de técnicas para reconhecimento de padrões; b) desenvolver uma técnica de reconhecimento de padrões baseada no cálculo de distâncias; c) comparar a técnica, aqui desenvolvida, com algumas técnicas tradicionais; d) finalmente, desenvolver um software genérico de reconhecimento de padrões, utilizando técnicas – não só algumas tradicionais, mas também a aqui desenvolvida.

2 Fundamentação Teórica

Reconhecimento de padrões – o ato de pegar dados brutos e agir baseado na “categoria” do padrão – tem sido crucial para nossa sobrevivência; por mais de milhões de anos temos desenvolvido um sistema neural cognitivo altamente sofisticado para tais tarefas.

Duda, Hart e Stork, em *Pattern Classification*
[DUDA 2001]

Nosso cérebro tem grande capacidade de aprender a classificar; já as máquinas são bastante eficientes para realizar operações lógicas bem definidas. Surge a primeira dificuldade. Antes de tentar fazer com que máquinas reconheçam padrões é importante entender como fazemos. Ao olhar para os símbolos mostrados na Figura 2.1, em todas as linhas é possível identificar o alfabeto. Na primeira coluna, todos os sinais são facilmente entendidos como a primeira letra; nenhum deles é idêntico, mas existem algumas características comuns entre eles que permitem reconhecê-los como letra “A”, sem confusão possível com as demais letras do alfabeto.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>	<i>F</i>	<i>G</i>	<i>H</i>	<i>I</i>	<i>J</i>	<i>K</i>	<i>L</i>	<i>M</i>	<i>N</i>	<i>O</i>	<i>P</i>	<i>Q</i>	<i>R</i>	<i>S</i>	<i>T</i>	<i>U</i>	<i>V</i>	<i>W</i>	<i>X</i>	<i>Y</i>	<i>Z</i>

Figura 2.1 - Quatro caligrafias diferentes entre as diversas caligrafias possíveis para o alfabeto

No entanto, para que uma pessoa seja capaz de reconhecer as letras do alfabeto, deve aprender, antes, quais são os “padrões” de cada letra; ou melhor, conhecer as características que permitem identificar cada padrão. Uma pessoa alfabetizada deve ser capaz de, ao olhar para uma letra, extrair as características, identificá-la com um padrão e, por fim, classificá-la como uma das letras do alfabeto.

Ao olhar para a Figura 2.2, não é preciso ser alfabetizado para identificar uma garrafa e uma casa, mas é preciso saber, antes, o que é uma casa e o que é uma garrafa. Quando se olha para uma garrafa de verdade, não um desenho, esta também é reconhecida como uma garrafa. Logo, é possível extrair as mesmas características comuns entre o desenho de uma garrafa e a própria garrafa. Nem todo desenho de casa é igual, e nem todo desenho de garrafa é igual, mas o desenhista deve ser capaz de reproduzir as características básicas que permitam ao observador reconhecer os traços-padrão de uma garrafa ou de uma casa. Da mesma maneira, um observador deve ser capaz de olhar para o desenho, extrair as características, associá-lo ao padrão de um objeto conhecido e, por fim, classificá-lo.



Figura 2.2 - O desenhista reproduz as características básicas que permitam ao observador reconhecer os traços-padrão de uma garrafa ou de uma casa.

Entre esses dois processos naturais de reconhecimento de padrões já é possível notar ações comuns. O reconhecimento da fala também é semelhante: ao som de uma palavra, os ouvidos primeiramente extraem as características de frequência e, em seguida, o cérebro identifica os fonemas, reconhece como palavra e, por fim, classifica o seu significado. De modo similar, todo processo de reconhecimento de padrões deve extrair as características de determinado objeto, identificá-las com padrão conhecido e classificá-lo. *Característica* é uma grandeza extraída do objeto observado, que ajuda a identificá-lo. *Padrão* é o conjunto organizado de características escolhidas para identificar o objeto. Por sua vez, o padrão também deve possuir características semelhantes que permitam agrupá-las como pertencentes a uma classe. *Classe* é uma categoria de padrões.

2.1 Sistema de reconhecimento de padrões

A maioria dos sistemas de reconhecimento de padrões pode ser representada pelo diagrama de blocos da Figura 2.3 [MOREIRA 2002].

No bloco de *sensoriamento* as informações são “lidas” do objeto, por

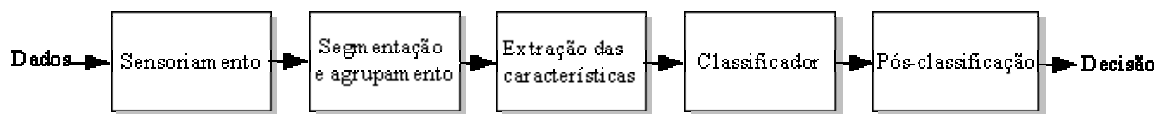


Figura 2.3 - Diagrama de blocos de um Sistema de Reconhecimento de Padrões

exemplo, utilizando-se uma câmera, um escâner, um microfone, etc.

A *segmentação e agrupamento* fazem a separação entre os padrões que são captados juntos pelo sensor. Por exemplo, separar as letras de uma palavra de um texto escaneado ou, mesmo, separar os fonemas e as palavras no reconhecimento de fala. Outra função desse bloco é fazer o agrupamento de padrões que têm partes separadas como, por exemplo, a letra “i”, composta por uma “barra” e um pingo.

Na *extração das características* procura-se fazer um pré-processamento para diminuir a dimensão do problema, captando as informações relevantes para a classificação do padrão. Dessa forma, o padrão é simplificado para um vetor de características. Por exemplo, extrair de um fonema as componentes de frequência, cada qual representada por uma dimensão do vetor. Duas técnicas bastante freqüentemente usadas para extrair características de sinais são as transformadas de Fourier e Wavelet [IAIONE 2002][ARGOUD 2001].

O *classificador* associa o vetor de características à categoria do padrão. Em reconhecimento de padrões essa tarefa nunca tem um desempenho perfeito, pois a eficiência depende da variabilidade dos valores das características dentro de uma categoria com relação aos valores das características de outras categorias. O bloco classificador é o principal foco de estudo deste trabalho; adiante, serão descritas algumas formas tradicionais de realizá-lo, como métodos estatísticos, redes neurais e, ainda, o novo método que aqui se está propondo.

O *pós-processamento* enquadra a classificação dentro de um contexto e analisa o risco da decisão. Na Figura 2.1, segunda linha, o caracter que parece ser a letra “O” na realidade foi feito com o caracter do zero (“0”) e possivelmente ao leitor envolvido no contexto, o detalhe passou despercebido. Mas, quais são as características que diferenciam a letra “O” do zero (“0”)?

Visualmente, somente o contexto diferencia; portanto, aquele zero é mesmo a letra “O”. Isso mostra a importância de se adequar a classificação ao contexto. O risco da decisão é importante em algumas aplicações; por exemplo, na classificação de um sinal de ECG de monitor cardíaco, em casos duvidosos, é melhor soar um alarme falso do que deixar passar uma fibrilação; em contrapartida, sendo excessivos, os alarmes falsos tornam o monitor sem confiabilidade.

2.2 Projeto de sistema de reconhecimento de padrões

Um projeto de sistema de reconhecimento de padrões começa pela coleta das amostras. Por exemplo, ao projetar um sistema que pretende classificar pessoas obesas e magras, é necessário ter um grupo de pessoas que sirvam de amostra. Cada pessoa deverá ser classificada por um especialista como obesa ou magra.

O próximo passo é a escolha das características que serão usadas para classificar pessoas como obesas ou magras. Nesse caso, as características mais evidentes são o peso e a altura. Mas uma pessoa com muita massa muscular poderia ser classificada de maneira errônea como obesa. Logo, peso e altura podem não ser suficientes para caracterizar uma pessoa como obesa ou magra; sendo assim, seria necessário procurar outra característica nas pessoas que melhore o sistema de classificação como, por exemplo, calcular o índice de massa corporal (peso/altura²).

O conjunto de características é chamado de vetor de características, em que cada característica é representada em uma dimensão do vetor. No exemplo, considerando apenas o peso e a altura, tem-se um vetor de características de duas dimensões. Dessa forma, pode-se considerar o problema de reconhecimento de padrões como uma função, cuja entrada é um vetor \mathbf{x} , de dimensão D , e a saída da função, um vetor \mathbf{y} de dimensão M . Usualmente, cada dimensão de \mathbf{y} pode variar de 0 a 1. O vetor de saída é associado a uma classe. O número de classes possível para se associar a saída \mathbf{y} é 2^M ou, então, o que é bastante comum na prática, é associar cada dimensão de saída y_m a uma

classe, ao invés de atribuir uma classe a uma combinação do vetor y . Neste caso, o número de classes de saída seria igual a M .

As etapas principais de um sistema de reconhecimento de padrões podem ser representadas pelo esquema da Figura 2.4.

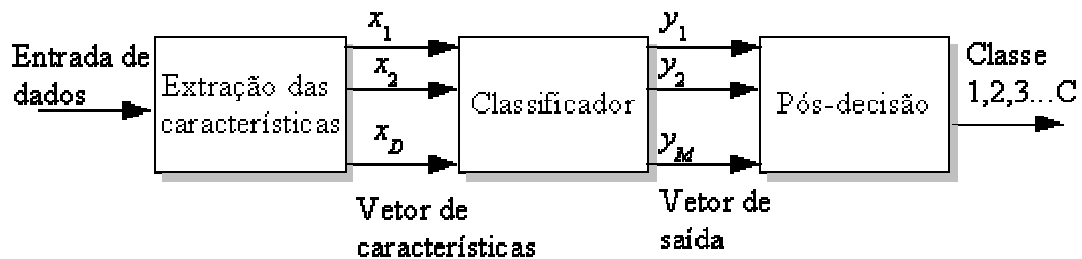


Figura 2.4 - Etapas principais de um sistema de reconhecimento de padrões

No capítulo 3 de “Pattern Classification” [DUDA 2001], é demonstrado que o incremento de uma característica (dimensão do vetor característica) pode reduzir a probabilidade de erro. Mas o número de combinações possíveis entre as características aumenta exponencialmente com o número de características. Supondo um problema cujas características são todas binárias, e D o número de características, o número de combinações entre características seria 2^D . Portanto, cada característica deve ser escolhida com muito critério. De maneira geral, quanto mais características, maior deve ser o conjunto de amostras [JAIN 2000]. Características mal escolhidas podem atrapalhar mais do que ajudar na classificação. Além disso, quanto mais características maior, será o custo computacional. Por essas razões, procura-se sempre diminuir o número de características.

Souza, em 1999, desenvolveu um sistema de reconhecimento de padrões, usando indexação recursiva (INREC), cujo principal objetivo é reduzir a dimensão das características. É importante saber o quão relevante é cada característica. Para extrair bem as características é necessário ter um profundo conhecimento do problema [SOUZA 1999].

A escolha do modelo utilizado refere-se principalmente ao bloco classificador (Figura 2.3). O classificador é a função que divide o espaço de características em regiões (volumes ou subespaços) que representam as classes

dos padrões. A *fronteira de decisão*, superfície que separa as classes no espaço de características, pode ter três formas básicas, ilustradas nos gráficos da Figura 2.5. A primeira forma, subtreinamento, só resolveria problemas muito simples, que fossem linearmente separáveis; como isso não ocorre em muitos casos reais, seu desempenho é, em geral, insatisfatório (Figura 2.5a). O segundo caso é exatamente o oposto: no sobretreinamento o classificador tem a fronteira de decisão exageradamente complexa (Figura 2.5b). Classifica perfeitamente todas as amostras de treinamento, mas, em geral, só tem desempenho satisfatório, quando o número de amostras disponíveis para o treinamento é relativamente pequeno. O sobretreinamento pode resultar de um excesso de interações durante a aprendizagem de uma rede neural perceptron multicamadas. Por último, em um treinamento adequado, o tipo de fronteira de decisão ideal não tem um desempenho perfeito para as amostras de treinamento, mas tem desempenho satisfatório para a classificação de novos padrões (Figura 2.5c).

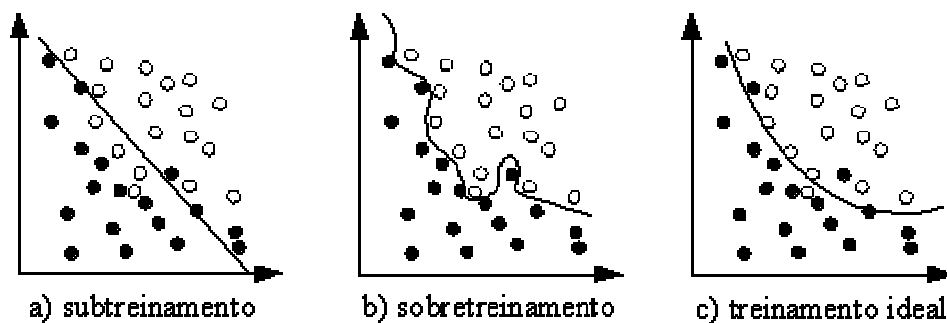


Figura 2.5 - Fronteiras de decisão: a) Muito simples, só resolve problemas lineares.

b) Muito complexa, tende a se especializar no conjunto de treinamento.

c) Tipo de fronteira de decisão desejada.

Se se utilizar os caracteres da Figura 2.1 (página 4) como conjunto de treinamento para um sistema de reconhecimento de caracteres, espera-se que seja capaz de reconhecer todo o alfabeto maiúsculo, mesmo que a fonte a ser reconhecida não seja exatamente qualquer uma das utilizadas no treinamento. Em suma: o sistema deve ser capaz de aprender a generalizar, classificando corretamente um padrão diferente dos apresentados no treinamento. Ter boa

capacidade de *generalização* é característica desejável no classificador.

Existem diversas maneiras de abordar reconhecimento de padrões.

A primeira abordagem, *comparação de modelos*, consiste em operações genéricas para determinar similaridades entre duas entradas (pontos, curvas ou contornos). Um padrão, para ser reconhecido, é comparado com os padrões armazenados (amostras) e é classificado na mesma classe do mais similar. Uma medida de similaridade freqüente é o cálculo de correlação.

Na segunda abordagem, *classificação estatística*, que é uma das mais estudadas na literatura, procura-se sempre saber qual é a probabilidade de um padrão pertencer a uma classe, associando a decisão a um risco de erro.

Algumas técnicas não exigem que exista um vetor de características com valores quantitativos, ou seja, valores reais. Esses métodos são considerados não métricos. Uma maneira, relativamente intuitiva, de reconhecer padrões é a de utilizar *árvore de decisão*. Nada mais é do que uma seqüência de procedimentos que testam as qualidades das características [DUDA 2001]. Em alguns casos, o que caracteriza um padrão é a inter-relação entre as características, ou seja, um conjunto estruturado de informações. Nesses casos utiliza-se a classificação *sintática ou estrutural* [JAIN 2000]. Para essa abordagem se faz uma descrição estrutural dos padrões na qual as classes são definidas pela similaridade estrutural entre os padrões. Normalmente se extraem, dos dados, primitivas (subpadrões) que descrevem padrões mais complexos, como na linguagem, em que as palavras fazem analogia com as primitivas, as frases seriam os padrões mais complexos e a gramática, as regras que classificam os padrões.

Por último, a abordagem *redes neurais*, atualmente uma das mais estudadas, tem inspiração biológica e é composta por vários neurônios artificiais interconectados com diferentes pesos sinápticos perfazendo uma rede.

As técnicas de reconhecimento de padrões podem ser *auto-organizáveis*, significando que basta ter um conjunto de amostras para que as regras de

classificação sejam encontradas de maneira autônoma. De outra forma, as regras para a classificação devem ser definidas pelo projetista nas outras técnicas. Por exemplo, uma rede neural perceptron com algoritmo de retropropagação é auto-organizável; já uma árvore de decisão não o é.

As técnicas auto-organizáveis e paramétricas* devem passar pelo processo de treinamento, podendo este ser ou não supervisionado. O *treinamento supervisionado* se dá, quando a classe de cada amostra do conjunto de treinamento é conhecida; já no *treinamento não supervisionado*, as classes das amostras não são conhecidas. Neste segundo caso, muitas vezes, o objetivo é descobrir se os dados podem ser classificados de alguma forma.

2.3 Complexidade Computacional

Para fazer comparação de algoritmos e de métodos computacionais, um parâmetro que deve ser levado em consideração é a complexidade computacional. Tal parâmetro está diretamente relacionado ao tempo e memória gastos para executar o algoritmo do método. Não faz muito sentido comparar métodos computacionais pelo tempo gasto para execução, pois isso varia muito de caso a caso, dependendo desde a maneira como estão estruturados os dados até a configuração do computador utilizado. Dizer simplesmente que um método tem complexidade maior que outro também é muito subjetivo. Por isso, para dar uma noção mais técnica e objetiva de complexidade computacional é preciso primeiro definir a ordem de uma função $f(x)$. Diz-se que $f(x)$ é de ordem de $h(x)$ representado por $f(x)=O(h(x))$ se existirem as constantes c e x_0 , tais que:

$$|f(x)| \leq c |h(x)| \text{ para todo } x > x_0 \quad (2.1)$$

A equação 2.1 significa que, para valores suficientemente altos de x em uma função, o crescimento da função $f(x)$ não é maior que $h(x)$.

Por exemplo, para a função $f(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2$ a ordem é $O(x^2)$ porque, com valores elevados de x , o termo constante e os termos $a_0 + a_1 \cdot x$ passam a ser sobrepostos pelo termo $a_2 \cdot x^2$, o que possibilita satisfazer a

* O conceito de técnica paramétrica e não paramétrica é esclarecido adiante no Capítulo 3.

equação 2.1.

Para descrever a complexidade computacional de um algoritmo, geralmente o que interessa são as operações matemáticas básicas: soma, subtração, multiplicação e divisão, enfim, o que se gasta de tempo e memória para ser processado [DUDA 2001]. Todas as operações que não são consideradas no cálculo da complexidade computacional são chamadas de *overhead*. O *overhead* é um parâmetro normalmente desprezível que só é considerado, quando a complexidade computacional entre dois algoritmos é muito próxima. Por exemplo, considerando dois algoritmos que possuem a mesma complexidade computacional, mas com *overhead* diferentes, aquele que tem maior *overhead* tem a maior complexidade computacional, mesmo que ambos sejam da mesma ordem.

3 Técnicas de Reconhecimento de Padrões

O diagrama da Figura 3.1 mostra os métodos de classificação que serão estudados junto com os métodos que estão sendo propostos aqui. Todos eles representam um padrão por um vetor de características. Os quadros em destaque representam os métodos que foram aplicados nesta pesquisa.

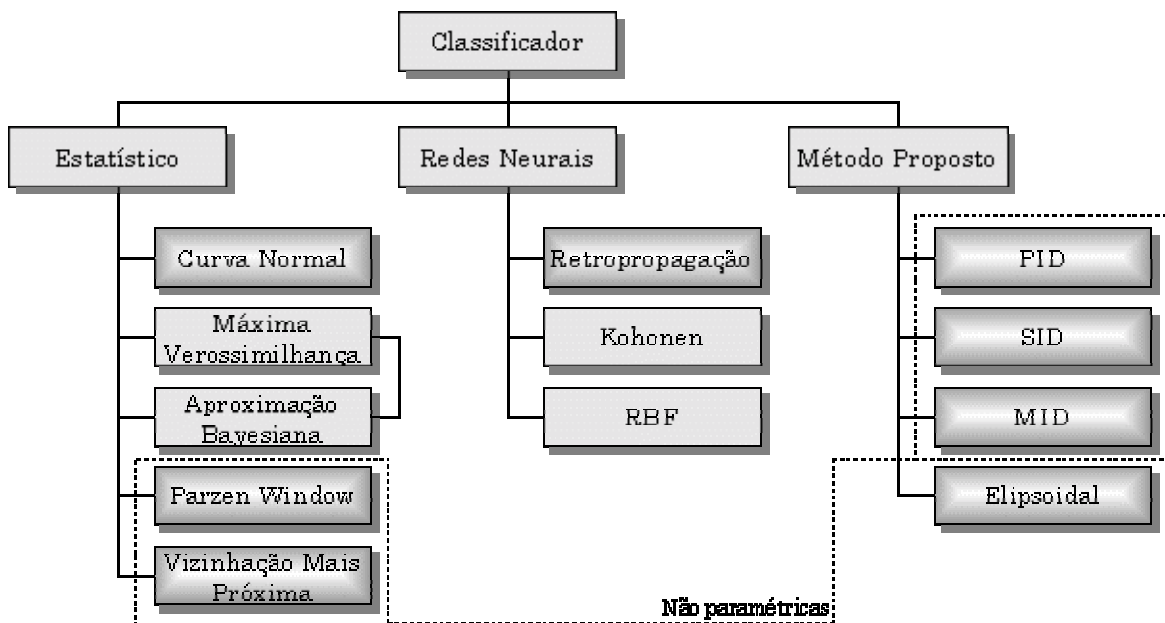


Figura 3.1 - Diagrama de blocos dos classificadores estudados e os propostos. O significado das siglas PID, SID e MID são respectivamente: Produto do Inverso das Distâncias, Somatório do Inverso das Distâncias e Mínima Distância.

3.1 Regra de Bayes

Pela regra de Bayes classifica-se um padrão, utilizando probabilidade. Calcula-se a probabilidade de o padrão pertencer a cada classe associando o risco de erro da escolha. Para se utilizar a regra de Bayes é importante ter uma estimativa das curvas de probabilidade de cada classe.

Considera-se *probabilidade a priori* (ou simplesmente *priori*) a frequência com que aparece cada classe. Sendo c uma classe, a probabilidade *a priori* é representada por $P(c)$. A soma das *prioris* de todas as classes deve ser 1.

Quando se tem somente a informação da probabilidade *a priori*, decide-se sempre por classificar um padrão com a classe que possui a maior *priori*, ou

seja, se a maioria das pessoas são obesas, classificar-se-iam todas as pessoas como obesas. Mas dificilmente tão pouca informação será utilizada para tomar uma decisão. No exemplo, pode-se passar a utilizar a informação do peso. Assim, a probabilidade de ocorrência de uma certa classe seria em função de uma informação (ou característica) conhecida, neste caso, o peso. Define-se, então, a curva de *densidade de probabilidade* de classe $p(x|c)$, é possível saber qual é a probabilidade *a posteriori*, dada uma informação (ex.: valor da característica peso). Probabilidade *a posteriori* $P(c|x)$ é a probabilidade de x pertencer à classe c . Exemplificando: a probabilidade *posteriori* $P(c_0|x)$ é a probabilidade de uma pessoa com o peso x pertencer à classe de obesos, c_0 .

$$P(c_j|x) = \frac{p(x|c_j) \cdot P(c_j)}{p(x)} \quad (3.1)$$

$$p(x) = \sum_{i=1}^c p(x|c_i) \cdot P(c_i)$$

Na regra de Bayes, equação 3.1, os termos relevantes para a classificação são $p(x|c_j)$ e $P(c_j)$. O fator $p(x)$ é o fator de *evidência*, que, para a classificação estatística, tem apenas a finalidade de ajustar a escala para que a soma de todas as probabilidades posteriores seja 1. Nos casos em que as probabilidades *a priori* são iguais, o único termo relevante é a densidade de probabilidade ($p(x|c_j)$).

Toda tomada de decisão envolve um risco de erro associado à escolha. O risco condicional da decisão pode ser calculado pela equação 3.2.

$$\mathfrak{R}(c_i|x) = \sum_{j=1}^c \lambda(c_i|c_j) \cdot P(c_j|x) \quad (3.2)$$

Onde $\lambda(c_i|c_j)$ é a *função perda*: prejuízo atribuído a classificação errônea, ou seja, quando se classifica o objeto como c_i enquanto o correto seria c_j . Uma forma comum de calcular a função perda é simplesmente atribuir 1 para as classificações erradas e 0 para as classificações corretas (equação 3.3).

$$\lambda(c_i|c_j) = \begin{cases} 0 & \text{se } i = j \\ 1 & \text{se } i \neq j \end{cases} \quad (3.3)$$

Nesse caso, o risco condicional fica equivalente à equação 3.4:

$$\mathfrak{R}(c_i|x) = 1 - P(c_i|x) \quad (3.4)$$

3.2 Distribuição normal

O desempenho da classificação bayesiana depende de como se estimam as curvas de densidade de probabilidade. Uma das formas de se fazer isto é utilizar a curva normal, muito conhecida na forma univariável (equação 3.5). No entanto, para aplicações em reconhecimento de padrões, a curva normal é necessária na forma multivariável (equação 3.6). Para estimá-la, calcula-se o vetor médio (equação 3.7) e a matriz de covariância (equação 3.8) das amostras pertencentes à mesma classe.

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right] \quad (3.5)$$

$$p(x) = \frac{1}{(2\pi)^{\frac{D}{2}} |\Sigma|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(x-\mu)^t \Sigma^{-1}(x-\mu)\right] \quad (3.6)$$

$$\mu = \frac{1}{N} \cdot \sum_{k=1}^N x_k \quad (3.7)$$

$$\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1j} \\ \sigma_{21} & \sigma_{22} & & \vdots \\ \vdots & & \ddots & \\ \sigma_{i1} & \cdots & & \sigma_{ij} \end{bmatrix} \quad (3.8)$$

$$\sigma_{ij} = \sum_{k=1}^n [(x_i - \mu_i) \cdot (x_j - \mu_j)]_k$$

A matriz covariância, equação 3.8, é sempre simétrica e positiva. O determinante de Σ deve ser sempre positivo. Os elementos σ_{ii} da diagonal

principal são os respectivos desvios-padrão dos parâmetros x_i , e os elementos σ_{ij} são as covariâncias entre os parâmetros x_i e x_j . Se $\sigma_{ij} = 0$, significa que os parâmetros x_i e x_j são independentes. De maneira geral, ao conhecer a matriz covariância, é possível calcular o grau de dispersão em qualquer direção ou subespaço [DUDA 2001].

3.3 Máxima Verossimilhança (Maximum Likelihood)

Diferentes tipos de equações podem ser utilizados para aproximar a densidade de probabilidade das classes de um sistema de reconhecimento de padrões. Para cada classe deverão ser encontrados diferentes parâmetros para a equação de aproximação utilizada. O método de aproximação por máxima verossimilhança é uma maneira de encontrar os parâmetros que melhor aproximam a densidade de probabilidade, com base em um conjunto de amostras. Por ser paramétrica, a densidade de probabilidade fica representada por $p(x|c, \theta)$, sendo θ o vetor de parâmetros. Cada classe é tratada independentemente uma da outra; ou seja, a densidade de probabilidade de uma classe não influencia na outra; assim, suas amostras são *i.i.d.*- variáveis aleatórias independentes e identicamente distribuídas. Por exemplo, no caso de uma aproximação feita por curva de distribuição normal, os elementos que preencheriam o vetor θ são a média e a matriz de covariância.

Sendo N_c o número de amostras da classe c_i e \mathcal{S} o conjunto de amostras referente à mesma classe, o cálculo da verossimilhança é feito pela equação 3.9.

$$p(c_i|\theta) = \prod_{k=1}^{N_c} p(\mathcal{S}_k|\theta) \quad (3.9)$$

Lê-se $p(\mathcal{S}_k|\theta)$ como a densidade de probabilidade de θ em função da amostra \mathcal{S}_k . Para encontrar os parâmetros que maximizam a verossimilhança, é intuitivo derivar a função em relação a θ e igualá-la a zero. No entanto, para facilitar a análise, é mais fácil trabalhar com o logaritmo da verossimilhança. Como o logaritmo é monotonicamente crescente, o θ que

maximizar o logaritmo da verossimilhança também maximizará a verossimilhança. Se $P(c_i|\boldsymbol{\theta})$ for uma função diferenciável em $\boldsymbol{\theta}$, os parâmetros poderão ser encontrados por métodos comuns de cálculo diferencial. Sendo p o número de parâmetros a ser estimado, então, faz-se $\boldsymbol{\theta}$ denotar um vetor de p componentes $\boldsymbol{\theta} = (\theta_1, \dots, \theta_p)^T$, e faz-se $\nabla_{\boldsymbol{\theta}}$ ser o operador gradiente (equação 3.10).

$$\nabla_{\boldsymbol{\theta}} = \begin{bmatrix} \frac{\delta}{\delta \theta_1} \\ \vdots \\ \frac{\delta}{\delta \theta_p} \end{bmatrix} \quad (3.10)$$

Define-se $l(\boldsymbol{\theta})$ como o logaritmo* da função verossimilhança (equação 3.11)

$$l(\boldsymbol{\theta}) \equiv \ln p(D|\boldsymbol{\theta}) \quad (3.11)$$

Sendo assim, para encontrar o $\boldsymbol{\theta}$ que maximize a verossimilhança, basta igualar o gradiente do logaritmo da verossimilhança a zero (equações 3.12 e 3.13).

$$l(\boldsymbol{\theta}) = \sum_{k=1}^n \ln p(\mathbf{x}_k|\boldsymbol{\theta}) \quad (3.12)$$

$$\nabla_{\boldsymbol{\theta}} l = 0 \quad (3.13)$$

A análise das soluções para essa equação deve ser feita com um certo cuidado para não confundir máximo global com máximos locais, mínimos locais e pontos de inflexão.

Para exemplificar, utiliza-se a curva normal univariável, supondo que as equações dos parâmetros (média e variância) não são conhecidas. Assim, $\theta_1 = \mu$ e $\theta_2 = \sigma^2$.

$$\ln p(x_k|\boldsymbol{\theta}) = -\frac{1}{2} \ln 2\pi\theta_2 - \frac{1}{2\theta_2} \cdot (x_k - \theta_1)^2 \quad (3.14)$$

derivando

* Normalmente se utiliza o logaritmo natural, mas pode ser em qualquer base; às vezes a escolha de uma base pode facilitar os cálculos.

$$\nabla_{\theta} l = \nabla_{\theta} \ln p(x_k | \theta) = \begin{bmatrix} \frac{1}{\theta_2} (x_k - \theta_1) \\ -\frac{1}{2\theta_2} + \frac{(x_k - \theta_1)^2}{2\theta_2^2} \end{bmatrix} \quad (3.15)$$

Aplicando-se a equação 3.13

$$\sum_{k=1}^n \frac{1}{\theta_2} (x_k - \theta_1) = 0 \quad (3.16)$$

e

$$\sum_{k=1}^n \frac{1}{\theta_2} + \sum_{k=1}^n \frac{(x_k - \theta_1)^2}{\theta_2^2} = 0 \quad (3.17)$$

Rearranjando as equações acima e fazendo $\theta_1 = \mu$ e $\theta_2 = \sigma^2$, obtém-se:

$$\mu = \frac{1}{N} \sum_{k=1}^N x_k \quad (3.18)$$

$$\sigma^2 = \frac{1}{N} \sum_{k=1}^N (x_k - \mu)^2 \quad (3.19)$$

3.4 Aproximação Bayesiana

As soluções pela aproximação bayesiana são quase sempre idênticas às do método de máxima verossimilhança, sendo a diferença mais conceitual do que prática. Enquanto no método da máxima verossimilhança procura-se fixar o valor do vetor de parâmetros, na aproximação bayesiana o vetor de parâmetros é considerado uma variável aleatória. Embora a densidade de probabilidade $p(x)$ seja desconhecida, considera-se que tem uma forma paramétrica conhecida $p(x|\theta)$. Assim, somente o vetor de parâmetros θ é considerado como desconhecido. O valor do vetor de parâmetros que representa a densidade de probabilidade em função do conjunto de amostras é $p(\theta | S)$.

$$p(\theta | S^N) = \frac{p(s_N | \theta) p(\theta | S^{N-1})}{\int p(s_N | \theta) p(\theta | S^{N-1}) d\theta} \quad (3.20)$$

A equação 3.20 é recursiva, onde $p(\theta | S^N)$ depende de $p(\theta | S^{N-1})$, sendo que $p(\theta | S^0) = p(\theta)$ deve ser estimado arbitrariamente. Mas o objetivo

de tudo isso é calcular a densidade de probabilidade $p(x)$ que pode ser estimada pela equação 3.21.

$$p(x|S) = \int p(x|\theta) p(\theta|S) d\theta \quad (3.21)$$

A análise dos resultados da aproximação bayesiana normalmente é mais complexa, comparada com a máxima verossimilhança. Isso porque, pela aproximação bayesiana, o resultado é uma distribuição de probabilidades do vetor de parâmetros, enquanto na máxima verossimilhança obtém-se um vetor de parâmetros. Quando se tem de optar entre os dois métodos, geralmente a máxima verossimilhança é escolhida, pois, além de ser mais fácil a interpretação dos resultados, pela máxima verossimilhança são necessários apenas cálculos de derivadas e gradientes, enquanto pela aproximação bayesiana é necessária uma integração multivariável.

3.5 Paramétrica X Não Paramétrica

As técnicas paramétricas, que utilizam a regra de Bayes, assumem que a função de densidade de probabilidade é conhecida e os parâmetros para a função são desconhecidos, mas podem ser estimados pelas amostras de treinamento. No entanto, para a maioria dos problemas de reconhecimento de padrões, essa suposição é no mínimo suspeita, pois dificilmente a forma de uma função de densidade de probabilidade se ajusta bem com a forma real. Além disso, todas as formas clássicas paramétricas são unimodais, enquanto a maioria dos problemas na prática são multimodais [JAIN 1988]. As técnicas não paramétricas podem ser usadas arbitrária e independentemente da distribuição e sem pressupor nenhuma forma de distribuição. Mas, naturalmente, haverá um custo computacional maior. As técnicas paramétricas não necessitam de treinamento, pois não há parâmetros nas equações que precisem ser ajustados em função das amostras. Mas todo o conjunto de treinamento é utilizado nos cálculos de reconhecimento. Aqui serão abordadas duas técnicas não paramétricas: Parzen Window e Nearest Neighbor.

3.6 Vizinhança mais Próxima (Nearest Neighbor)

O método da vizinhança mais próxima é um dos mais simples e intuitivo para estimar a probabilidade de \mathbf{x} pertencer a classe c . É comum abreviar-se o método como k-NN ou NN (Nearest Neighbor). A idéia básica já está dita no próprio nome, já que a probabilidade é definida pela vizinhança do ponto \mathbf{x} em questão. Este método estima diretamente a probabilidade *a posteriori* $P(c_i|\mathbf{x})$, sem a necessidade de estimar a densidade de probabilidade. Procurando, entre as amostras, as k amostras mais próximas de \mathbf{x} , levando em consideração as amostras de todas as classes. Sendo k_i o número de amostras encontrado da classe c_i e a probabilidade *a posteriori* $P(c_i|\mathbf{x})$ a razão k_i/k .

$$P(c_i|\mathbf{x}) = \frac{k_i}{k} \quad (3.22)$$

A densidade de probabilidade pode ser estimada pela seguinte equação:

$$p(\mathbf{x}|c_i) = \frac{k_i/N}{V} \quad (3.23)$$

sendo V o volume que envolve as k amostras capturadas.

Idealmente, para que a classificação k-NN seja perfeita, $N \rightarrow \infty$ e $k \rightarrow \infty$ [DUDA 2001][HART 1967].

Toda a base do cálculo da vizinhança mais próxima depende de como se define distância. As três formas mais comuns de definir distância são: Manhattan, Euclideana e Mahalanobis.

Manhattan:

$$D_{MH}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^D |x_i - y_i| \quad (3.24)$$

Essa é umas das formas mais simples e rápidas de calcular distância, mas não fornece os melhores resultados na classificação. A vantagem é ser executada com maior rapidez que a euclideana e a Mahalanobis. Mantendo a mesma distância Manhattan de um determinado ponto em duas dimensões obtém-se um quadrado.

Euclideana:

$$D_E(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^D (x_i - y_i)^2} \quad (3.25)$$

É o modo mais comum de calcular distância. Em duas dimensões é equivalente ao teorema de Pitágoras. Para a maioria dos casos, tem boa eficiência. Mantendo a mesma distância euclideana de um determinado ponto em três dimensões, obtém-se uma esfera.

Mahalanobis:

$$D_{ML}(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^D (x_i - \mu_i)^T \Sigma^{-1} (y_i - \mu_i)} \quad (3.26)$$

Sendo μ e Σ a média e a matriz de covariância respectivamente, esta acaba sendo uma forma complexa de calcular distância, pois, se a dimensão D de entrada for alta, a complexidade computacional para inverter a matriz covariância aumentará muito. A vantagem da distância Mahalanobis é que esta leva em consideração a distribuição estatística das amostras e resolve alguns problemas que podem ser encontrados nos outros cálculos de distância como características com escalas mal dimensionadas. Por exemplo, as características peso e altura devem ter o mesmo fator de escala? Outro problema que pode ser minimizado com a distância Mahalanobis é quando se têm características altamente correlacionadas. Devido à alta complexidade computacional deste cálculo de distância, seu uso é pouco vantajoso e comum.

A Figura 3.2 ilustra a diferença entre os cálculos de distâncias.

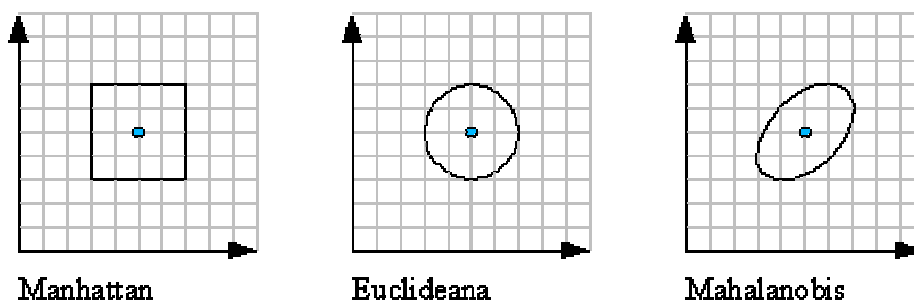


Figura 3.2 - Representação gráfica dos cálculos das diferentes distâncias

A principal limitação do método da vizinhança mais próxima é o custo computacional. A princípio, para encontrar as k amostras mais próximas de \mathbf{x} ,

é necessário calcular a distância entre \mathbf{x} e todas as n amostras. Bandyopadhyay [BANDYOPADHYAY 2001] propõe um pré-processamento das amostras, fazendo uma reordenação que otimiza a busca pelas amostras mais próximas. Wu propõe um algoritmo que reduz o conjunto de amostras sem perder precisão de classificação [WU 2001].

3.7 Parzen Window

Enquanto pelo método da vizinhança mais próxima se aumenta uma janela ao redor de \mathbf{x} até que esta envolva k amostras, na aproximação Parzen Window a idéia é fixar uma janela ao redor de \mathbf{x} e verificar quantas amostras existem dentro.

A janela é definida por um hipercubo de arestas h e volume V , (equação 3.27).

$$V = h^D \quad (3.27)$$

Para encontrar quantas amostras caem dentro do volume V define-se a função janela pela equação 3.28.

$$\varphi(\mathbf{u}) = \begin{cases} 1 & |u_j| \leq \frac{1}{2}; j, \dots, D \\ 0 & \text{caso contrário} \end{cases} \quad (3.28)$$

Essa função define um hipercubo de volume 1 centrado na origem. A função janela pode ser definida de outra maneira, mas obedecendo à condição da equação 3.29.

$$\int_{-\infty}^{\infty} \varphi(\mathbf{u}) d\mathbf{u} = 1 \quad (3.29)$$

A função que define quantas amostras estão dentro da janela de volume V centrada em \mathbf{x} é dada pela equação 3.30:

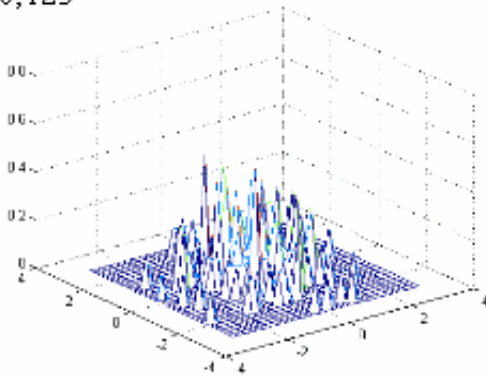
$$k_i = \sum_{j=1}^N \varphi\left(\frac{\mathbf{x} - \mathbf{x}_j}{h}\right) \quad (3.30)$$

Assim, a densidade $p(\mathbf{x})$ pode ser estimada pela equação 3.31:

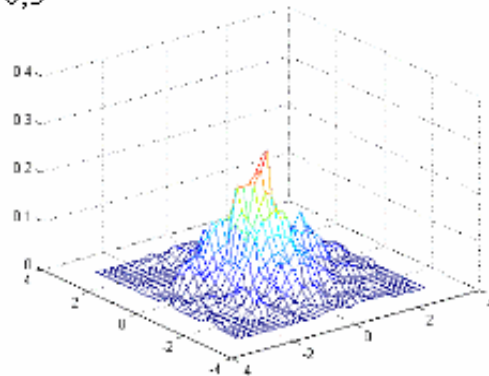
$$p(\mathbf{x}) = \frac{1}{N} \sum_{j=1}^N \frac{1}{V} \varphi\left(\frac{\mathbf{x} - \mathbf{x}_j}{h}\right) \quad (3.31)$$

Quanto maior for o número de amostras (n) e menor for o valor de h mais próximo da real densidade de probabilidade irá se aproximar $p(\mathbf{x})$. Como n é limitado, na prática h nunca poderá tender a zero. Valores muito altos para h fazem $p(\mathbf{x})$ ter baixa resolução, já valores excessivamente baixos fazem a densidade de probabilidade ter variabilidade muito alta. Com um número limitado de amostras é preciso procurar um valor para h que não comprometa a resolução e que não tenha uma excessiva variabilidade. Os gráficos da Figura 3.3 ilustram curvas de densidade de probabilidade estimada por Parzen Window com diferentes valores de h . Para esta simulação foram geradas 500 amostras para uma distribuição normal de duas dimensões com a média na origem e desvio-padrão 1.

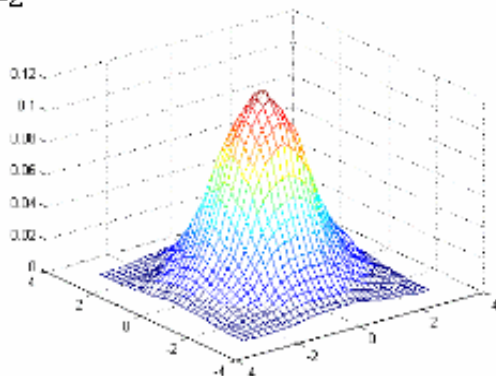
$h=0,125$



$h=0,5$



$h=2$



$h=8$

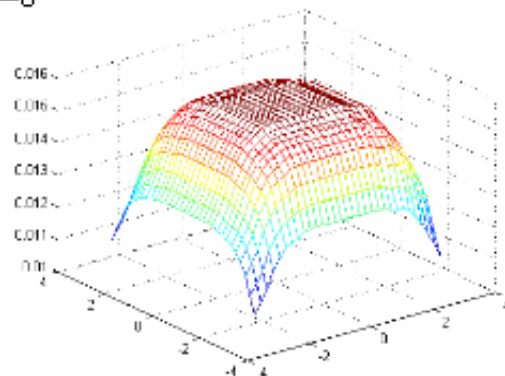


Figura 3.3 - Simulação Parzen window com diferentes valores de h

Note, nos gráficos, que a escala do eixo z varia para cada um. Quanto maior for o valor de h , menores serão os valores absolutos de $p(\mathbf{x})$ e vice-versa.

No gráfico, onde $h=0,125$, fica nítido o excesso de variabilidade, enquanto onde $h=8$, percebe-se uma perda grande de resolução, principalmente em torno da origem.

3.8 k-Médias

K-médias (ou k-means) é um algoritmo de aglomeração (clustering), cujo objetivo é reduzir um grupo de amostras a uma única amostra que represente bem todas as amostras aglomeradas. Pode ser aplicado no reconhecimento por k-NN, Parzen window e no treinamento de redes neurais. As técnicas de aglomeração também podem ser usadas para fazer reconhecimento de padrões não supervisionado, onde cada aglomerado passa a ser uma classe.

O número de aglomerações formadas é previamente definido pelo valor de k . Cada aglomeração é representada pelo valor de um vetor de dimensão d igual ao vetor de características. A seqüência de instruções para o k-médias é a seguinte:

1. Escolha um valor de k conveniente para o problema.
2. Crie aleatoriamente k vetores μ de dimensão d que serão os centros provisórios dos aglomerados.
3. Associe cada amostra com o aglomerado μ_k mais próximo.
4. Calcule o vetor médio das amostras de cada aglomerado.
5. Faça μ_k igual a seu respectivo valor médio.
6. Volte ao passo 3 até que não haja mais alterações nos aglomerados.

Para o passo 3 é preciso calcular distâncias, sendo que normalmente a utilizada é a euclideana, mas pode-se utilizar qualquer outra.

3.9 Redes Neurais

Em 1943, o fisiologista Warren S. McCulloch e o matemático Walter H. Pitts modelaram o neurônio natural como um circuito binário. Pitts, usando argumentos lógicos, provou a equivalência de uma rede neural com a máquina de Turing. A partir daí, havia dois motivos para estudar redes neurais:

modelar o sistema nervoso de um ser vivo e construir computadores com alto grau de paralelismo [de AZEVEDO 2000].

Provavelmente, devido à modelagem das redes neurais biológicas ser uma tarefa extremamente complexa, ainda hoje, as redes neurais artificiais pouco se assemelham com as elas; no entanto, diversas topologias de redes foram desenvolvidas com vastas aplicações.

Redes neurais podem ser vistas como um sistema de computação paralela composto por vários processadores simples interconectados. Mesmo existindo a possibilidade de se implementar uma rede neural em hardware, modelando-se eletronicamente cada neurônio, na maioria das aplicações tem sido suficiente modelar toda a rede em software e executá-la em processador único.

Entre as diversas topologias de redes neurais, as mais comuns usadas na tarefa de classificação de padrões têm sido as redes perceptron multicamadas, redes RBF (Radial-Basis Function) e a rede de Kohonen [JAIN 2000].

3.9.1 Modelo de Neurônio

Um neurônio é composto, basicamente, por um corpo celular, dendritos e axônio. Os dendritos são as entradas do sistema e o axônio, a saída. Um neurônio pode possuir muitas entradas, mas apenas uma saída. Um axônio pode se ligar a vários dendritos de outros neurônios, formando, então, uma rede de neurônios. Cada ligação entre axônio e dendrito tem um peso sináptico associado. A entrada da rede é feita diretamente nos dendritos dos neurônios, tendo também um peso sináptico associado. A saída é tirada dos axônios dos neurônios.

De Azevedo [de AZEVEDO 2000] propõe um modelo geral de neurônio (em 1993) que engloba todas as possibilidades de modelagem. No entanto, para este trabalho será utilizado um modelo mais simples (veja Figura 3.4), onde a saída do neurônio é uma função de ativação, cujo resultado depende do somatório das entradas (dendritos), lembrando que cada entrada é ponderada

pelo seu peso sináptico associado.

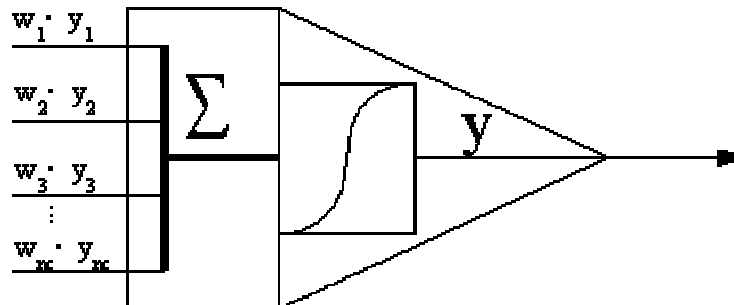


Figura 3.4 - Neurônio artificial

3.9.2 Perceptrons de múltiplas camadas

A rede perceptron de múltiplas camadas é uma rede direta, ou seja, a ativação dos neurônios se propaga em apenas uma direção, não havendo realimentação (Figura 3.5). O número de camadas de uma rede é livre; no entanto, para resolver problemas não lineares, deve-se ter pelo menos duas camadas de pesos adaptativos [de AZEVEDO 2000]. No entanto, um número de camadas muito elevado pode aumentar a complexidade da rede sem melhorar o desempenho.

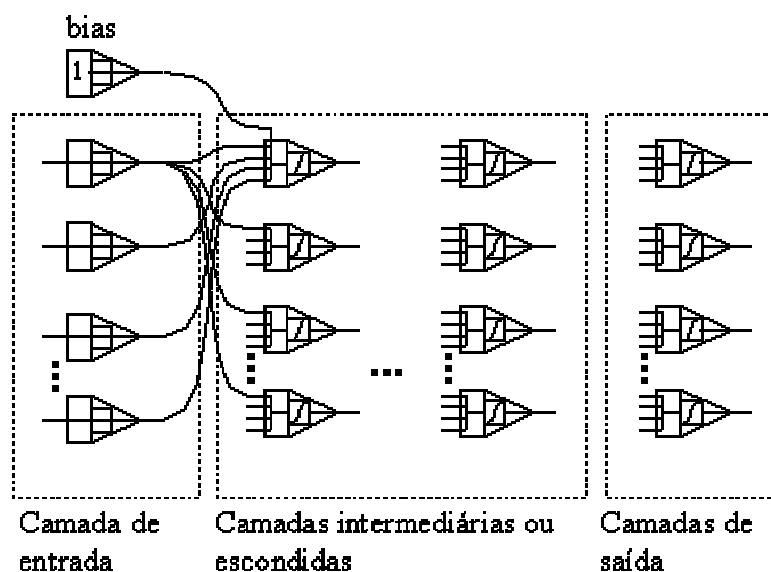


Figura 3.5 - Rede perceptron de múltiplas camadas

Na camada de entrada da rede os neurônios são ativados pelo vetor de

características. A ativação dos neurônios se propaga até a camada de saída. A camada de saída pode ser projetada para representar as classes de duas formas. Da primeira maneira, cada neurônio da camada de saída representa uma classe. Desse modo, para representar oito classes, a camada de saída deve ter oito neurônios. Do segundo modo, cada classe é representada pela combinação binária das saídas. Assim, com apenas oito neurônios é possível representar duzentos e cinquenta e seis classes.

O algoritmo típico de treinamento da rede perceptron multicamadas é o de retropropagação, também conhecido como regra delta generalizada. Esse algoritmo faz o ajuste dos pesos adaptativos de acordo com o conjunto de amostras. Resumidamente o algoritmo de treinamento por retropropagação consiste no seguinte [de AZEVEDO 2000]:

- 1 Inicia pesos com valores aleatórios.
- 2 Seleciona amostras.
- 3 Propaga ativação dos neurônios com a entrada da amostra.
- 4 Calcula erro da saída dos neurônios comparando com a saída esperada pela amostra.
- 5 Calculam erros das camadas intermediárias em função do erro da camada seguinte.
- 6 Corrige valores dos pesos adaptativos em função dos respectivos erros.
- 7 Volta ao passo 2.

Este algoritmo repete até que o erro fique abaixo de um patamar estabelecido, ou até que exceda um certo número de épocas de treinamento (temos uma época quando todas as amostras do conjunto de treinamento já foram selecionadas).

Há diversas variações do algoritmo de treinamento de retropropagação que procuram agilizar e melhorar o treinamento.

Neste trabalho, o algoritmo de retropropagação utilizado foi o mais simples, com a função de ativação sigmoïdal da equação 3.32.

$$y = \frac{1}{1 + e^{-g \cdot u}} \quad (3.32)$$

Sendo u o somatório das entradas do neurônio ponderado pelos pesos sinápticos e g um escalar positivo, quanto maior for o valor de g , maior a inclinação de y .

3.9.3 Redes RBF

A rede de função de base radial ou RBF (Radial Base Function), em seu modo mais simples, constitui-se em uma rede direta de três camadas, semelhante à rede perceptron (Figura 3.6).

A primeira camada, assim como em uma rede perceptron, só repassa o valor do vetor de entrada para a camada intermediária. A camada intermediária faz a transformação linear, geralmente aumentando a dimensão do espaço vetorial de entrada para um espaço vetorial maior interno. Os neurônios desta camada é que são funções radiais de base, dando nome à rede.

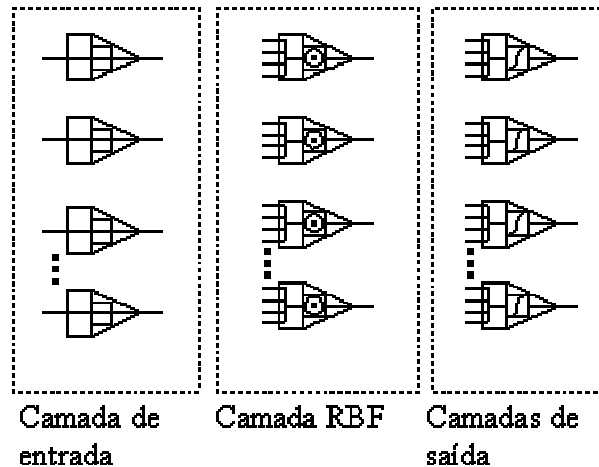


Figura 3.6 - Rede RBF

Já a terceira camada é semelhante à camada de saída de uma rede perceptron, sendo a saída de cada neurônio uma transformação linear em função do somatório das entradas ponderadas pelos pesos das conexões entre as camadas.

A função radial de base possui um centro (ce) e um raio, de forma que, quando a entrada está afastada do centro, a saída tende a ser praticamente

nula, do contrário tende a um (1). Uma das funções mais comuns, usadas como função de ativação, é a função de Gauss (equação 3.33) [TODESCO 1995].

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{D/2} \sigma_1 \sigma_2 \sigma_3 \cdots \sigma_D} \exp\left(-\frac{1}{2} \sum_{j=1}^D \left(\frac{x_j - ce_j}{\sigma_j}\right)^2\right) \quad (3.33)$$

Os valores $\sigma_1 \sigma_2 \sigma_3 \cdots \sigma_D$ são usados da mesma maneira que na distribuição normal, nada mais sendo que o grau de dispersão ao redor de \mathbf{ce} . Note que o termo dentro do somatório é equivalente ao cálculo da distância euclidiana ponderado por um fator de escala σ . Para aumentar a funcionalidade da função de base pode ser utilizado o cálculo de distância Mahalanobis no lugar da euclidiana.

Uma maneira comum de treinar redes RBF é usar um treinamento híbrido, sendo a primeira parte não supervisionada e a segunda parte, supervisionada. No primeiro momento usa-se uma técnica de aglomeração, normalmente k-médias, para se encontrar o centro da função. Uma maneira simples de se estimar o vetor σ é fazer a média das distâncias das amostras. Encontrados os parâmetros das funções radiais, o próximo passo é fazer o treinamento supervisionado com retropropagação.

Uma rede perceptron pode equivaler a uma rede RBF e vice-versa. No entanto, a rede RBF comum possui apenas três camadas, enquanto a perceptron pode possuir várias. O treinamento da RBF tende a ser mais rápido, mas normalmente a classificação (propagação) da rede perceptron é mais rápida. A rede RBF possui dois modelos de neurônios, enquanto a perceptron pode ser toda modelada com modelo único de neurônio. Uma rede perceptron tende a mapear todo espaço de entrada, generalizando até para locais onde há pouca ou nenhuma amostra, enquanto as redes RBF decrescem exponencialmente as aproximações, quando está longe dos centros das funções RBF.

3.9.4 Rede Kohonen

A rede Kohonen, também conhecida como mapa de Kohonen ou SOM (Self Organization Map), é usada para reconhecimento de padrão não

supervisionado e aglomeração (clustering). A estrutura básica dessa rede é composta por duas camadas: a primeira, assim como nas outras redes, é a camada de entrada composta por D neurônios que só têm a função de repassar o valor do vetor de entrada para a próxima camada. A segunda camada é uma matriz bidimensional quadrada (algumas vezes utiliza-se um vetor unidimensional) de neurônios competitivos (Figura 3.7).

No caso mais geral, todos os neurônios da camada competitiva estão conectados na camada de entrada com conexões excitatórias, e entre os neurônios da camada competitiva há conexões inibitórias. O valor dos pesos é limitado entre 0 e 1 e o somatório dos pesos de uma unidade deve ser 1 (equação 3.34).

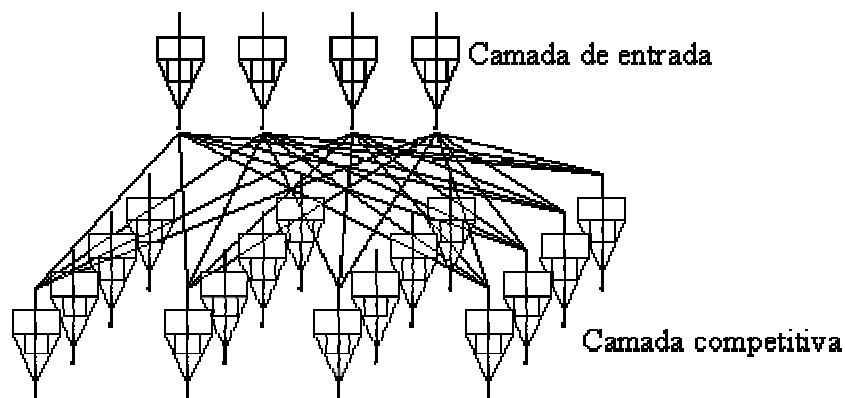


Figura 3.7 - Rede Kohonen (SOM)

$$\sum_i w_{ij} = 1 \quad (3.34)$$

Iniciam-se os pesos de forma aleatória, satisfazendo a equação 3.34. O neurônio típico da camada competitiva é definido pela equação 3.35.

$$y_i = \begin{cases} +1 & \text{se } (u_i > u_k) \text{ para todo } k \\ 0 & \text{caso contrário} \end{cases} \quad (3.35)$$

O neurônio vencedor ganha tudo, ou seja, o neurônio que, pela equação 3.35, resultar em 1, terá seus pesos atualizados pela equação 3.36.

$$\Delta w_{ij} = \eta \left(\frac{u_i}{n} - w_{ij} \right) \quad (3.36)$$

Onde: η é a constante de aprendizagem, n o número das unidades ativas e $u_i = 1$, se o estímulo da unidade i da camada anterior estiver ativo, do contrário $u_i = 0$. Analisando-se as equações de treinamento, nota-se que o neurônio aprende, deslocando os pesos de suas conexões das entradas inativas para as entradas ativas.

4 Técnicas Propostas

Todas as técnicas de reconhecimento de padrões que aqui serão propostas baseiam-se em cálculo de distâncias. O padrão é classificado pela comparação de suas distâncias com cada amostra, cuja classe já é conhecida. De modo simples, pode-se dizer que a classe do padrão é determinada pelas amostras mais próximas. Foram definidas basicamente três formas de determinar a proximidade:

1. Produto do inverso das distâncias (PID).
2. Somatório do inverso das distâncias (SID).
3. O inverso da menor distância (mínima distância - MID).

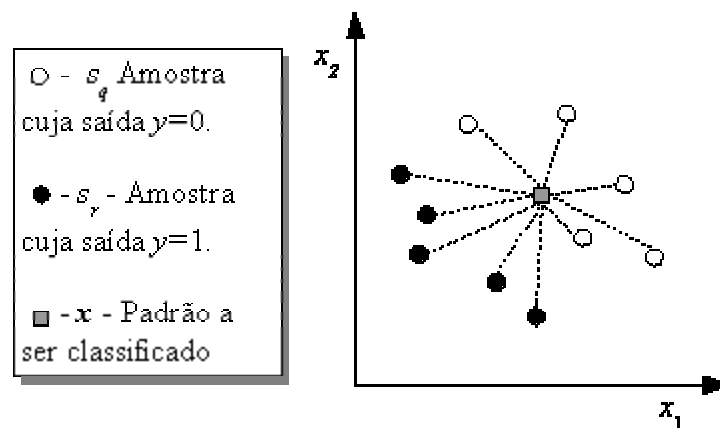


Figura 4.1 - Ilustração dos métodos PID, SID e MID.

A Figura 4.1 procura ilustrar o funcionamento dos métodos propostos. Os três métodos (PID, SID e MID) precisam calcular todas as distâncias entre o padrão a ser classificado e cada amostra. As distâncias são separadas em dois grupos, um das amostras de saída 0 e outro das amostras de saída 1. Para o cálculo do MID, separa-se a menor distância entre as amostras de saída 1 ($\min(\text{dist}(s_r, \mathbf{x}))$) e a menor distância entre as amostras de saída 0 ($\min(\text{dist}(s_q, \mathbf{x}))$), inverte-se os resultados e subtrai-se um do outro, conforme a equação 4.1:

$$T = \frac{1}{\min(\text{dist}(s_r, \mathbf{x}))} - \frac{1}{\min(\text{dist}(s_q, \mathbf{x}))} \quad (4.1)$$

Note-se que, quando a menor distância entre as amostras de saída 1

tender a 0, ou seja, quando \mathbf{x} estiver muito próximo de uma amostra cuja saída é 1, T tenderá a $+\infty$; do contrário, se a menor distância entre as amostras cuja saída é 0 tender a 0, T tenderá a $-\infty$. Caso as duas distâncias sejam muito semelhantes, T tenderá a 0. Sendo assim, o resultado de T pode ser aplicado a uma equação sigmoidal como a equação 4.2, onde g é um escalar positivo.

$$y = \frac{1}{1 + e^{-g \cdot T}} \quad (4.2)$$

Quando T tender a $+\infty$, y tenderá a 1 e quando T tender a $-\infty$, y tenderá a 0. Caso T tenda a 0, y tenderá a 0,5. Ou seja, quando \mathbf{x} estiver bem próximo a uma amostra de saída 1, y ficará entre 0,5 e 1 e, se \mathbf{x} estiver bem próximo a uma amostra de saída 0, y ficará entre 0 e 0,5.

O método PID funciona de modo semelhante; no entanto, ao invés de calcular a menor distância, faz-se o produtório do inverso das distâncias, conforme equação 4.3.

$$T = \prod_{q=0}^Q \frac{1}{\text{dist}(s_q, \mathbf{x})} - \prod_{r=0}^R \frac{1}{\text{dist}(s_r, \mathbf{x})} \quad (4.3)$$

Pelo método SID a equação é bastante semelhante, trocando-se, apenas, o produtório por um somatório, conforme equação 4.4.

$$T = \sum_{q=0}^Q \frac{1}{\text{dist}(s_q, \mathbf{x})} - \sum_{r=0}^R \frac{1}{\text{dist}(s_r, \mathbf{x})} \quad (4.4)$$

Uma análise semelhante feita para a equação 4.1 pode ser realizada para as equações 4.3 e 4.4, mostrando que, quando \mathbf{x} está muito próximo de uma das amostras de saída 1, T tenderá a $+\infty$, quando \mathbf{x} está muito próximo de uma das amostras de saída 0, T tenderá a $-\infty$ e, por fim, se \mathbf{x} estiver a uma distância média semelhante para os dois grupos de amostras, T tenderá a 0.

Note-se que, por qualquer um dos métodos, se \mathbf{x} coincidir exatamente com uma das amostras de saída 1, y será exatamente 1 e, se \mathbf{x} coincidir exatamente com uma das amostras de saída 0, y será exatamente 0.

4.1 Formalizando e refinando as técnicas propostas

A proposta é desenvolver um classificador com D características de entrada e M saídas, sendo que o número possível de classes para associar a saída é 2^M . O vetor de saída é representado por \mathbf{y} e tem dimensão M . Cada dimensão de \mathbf{y} tem valor variando de 0 a 1. Na prática, é mais fácil e comum associar cada dimensão de saída y_m a uma classe, ao invés de atribuir uma classe a uma combinação do vetor \mathbf{y} . Nesse caso, o número de classes de saída seria igual a M . Os métodos PID, SID e MID não são paramétricos, pois não há parâmetros a serem ajustados pelo conjunto de amostras.

Todos os métodos propostos utilizam a equação 4.5, mudando apenas o cálculo de T_m . O escalar positivo g tem um significado importante, pois ele controla o grau de generalização da classificação. Note-se que, quando g tende a 0, y_m tende a 0,5 e quando g tende a ∞ , y_m tende a 1, se T for positivo, e a 0, se T for negativo. Ou seja, se g tiver um valor baixo, y_m só tenderá a uma das classes, quando \mathbf{x} estiver já bem próximo a uma das amostras (baixo grau de generalização). Já se g tiver um valor alto, y_m tenderá sempre a qualquer uma das classes, mesmo que \mathbf{x} esteja relativamente longe de qualquer uma das amostras (alto grau de generalização). Não foi determinado nenhum método específico para se encontrar um valor ideal para g , pois o método pode ser particular para cada tipo de problema. No entanto, fica como sugestão arbitrar valores para g e, por interpolação, procurar g que minimize o erro quadrático médio da classificação das amostras de teste.

$$y_m = \frac{1}{1 + e^{-g \cdot T_m}} \quad (4.5)$$

Considerando que a fronteira de decisão entre as classes (página 9) está onde $y_m=0,5$, ou seja, $T_m=0$, pode-se fazer uma análise mais profunda com relação a g . Nas fronteiras de decisão α é o ângulo formado entre y_m e T_m , sendo que $\alpha = \arctg(g/4)$. Quanto menor o valor de α , mais suave é a transição entre as classes e, quanto maior for α , mais brusca é a transição. O que significa que, quanto maior for o valor de g , maior será o ângulo de

inclinação na fronteira de decisão entre as classes (equações 4.6 e 4.7).

$$tg \alpha = \frac{dy_m}{dT_m} = \frac{g \cdot e^{-g \cdot T_m}}{1 + 2 \cdot e^{-g \cdot T_m} + e^{-2g \cdot T_m}} \quad (4.6)$$

$$\lim_{T_m \rightarrow 0} tg \alpha = \frac{g}{4} \quad (4.7)$$

As diferentes formas de calcular T_m sempre se baseiam no cálculo de distância entre as amostras. Teoricamente, qualquer maneira de calcular distância poderia ser utilizada. Todavia, neste trabalho resultou a conveniência de normalizar o cálculo da distância para diferentes dimensões. Pela equação 4.8, toda distância medida da origem a um vetor unitário resulta 1, independentemente da dimensão do vetor. O significado prático é que fica mais fácil ajustar o valor de g para problemas de altas dimensões; senão, bons valores de g tenderiam a ser cada vez menores, conforme aumentasse D , enquanto, com a distância normalizada, um valor ideal de g fica sempre em torno de 1.

$$dist(\mathbf{a}, \mathbf{b}) = \sqrt{\frac{\sum_{d=0}^D (a_d - b_d)^2}{d}} \quad (4.8)$$

4.2 PID (Produtório do Inverso das Distâncias)

As equações 4.9 e 4.10 utilizadas para o cálculo do método PID são equivalentes, quando $Q=R$, onde Q é o número de amostras da saída m , cujo valor de saída é 1, e R é o número de amostras da saída m cujo valor de saída é 0. Se $Q \neq R$, as equações diferem; a equação 4.9 considera, de certa forma, a probabilidade *a priori*, já a equação 4.10 não. Por consequência, utilizando a equação 4.9, quando $Q \gg R$, a saída y tenderá a 1; de maneira inversa, se $Q \ll R$, a saída y tenderá a 0. No entanto, utilizando a equação 4.10, a probabilidade de y ser 0 ou 1 vai depender apenas da distribuição das amostras e não da diferença de amostras para cada classe.

$$T_m = \sqrt{\prod_{q=0}^Q \frac{1}{dist(s_q, \mathbf{x})} - \prod_{r=0}^R \frac{1}{dist(s_r, \mathbf{x})}} \quad (4.9)$$

$$T_m = \sqrt[Q]{\prod_{q=0}^Q \frac{1}{\text{dist}(s_q, \mathbf{x})}} - \sqrt[R]{\prod_{r=0}^R \frac{1}{\text{dist}(s_r, \mathbf{x})}} \quad (4.10)$$

Observa-se que s_q subentende-se como as amostras cujo valor de saída é 1 e s_r cujo valor de saída é 0.

4.3 SID (Somatório do Inverso das Distâncias)

As equações para o cálculo do SID são análogas às equações para o cálculo do PID, trocando-se, apenas, produtório por somatório e raiz por divisão. O “2” que aparece multiplicando a equação 4.11 é uma constante de proporcionalidade que serve somente para tornar as equações 4.11 e 4.12 equivalentes, quando $Q = R$. Da mesma maneira que nas equações do PID, a equação 4.11 considera a probabilidade *a priori* e a equação 4.12, não.

$$T_m = 2 \cdot \frac{\sum_{q=0}^Q \frac{1}{\text{dist}(s_q, \mathbf{x})} - \sum_{r=0}^R \frac{1}{\text{dist}(s_r, \mathbf{x})}}{N} \quad (4.11)$$

$$T_m = \frac{\sum_{q=0}^Q \frac{1}{\text{dist}(s_q, \mathbf{x})}}{Q} - \frac{\sum_{r=0}^R \frac{1}{\text{dist}(s_r, \mathbf{x})}}{R} \quad (4.12)$$

4.4 MID (Mínima Distância)

O método MID, representado pela equação 4.13, considera somente a amostra mais próxima de cada classe, desprezando totalmente as outras.

A resposta é muito semelhante ao 1-NN, com a diferença de que no 1-NN a saída é binária e, no caso do MID, a saída é contínua entre 0 e 1. A semelhança possibilita aplicar todos os métodos desenvolvidos para otimizar k-NN no método MID [BANDYOPADHYAY 2001][WU 2001].

$$T_m = \frac{1}{\min(\text{dist}(s_Q, \mathbf{x}))} - \frac{1}{\min(\text{dist}(s_R, \mathbf{x}))} \quad (4.13)$$

4.5 Elipsoidal

Os métodos PID e SID usam, necessariamente, todas as amostras para fazer a classificação. No método MID, se não for utilizado nenhum pré-

processamento para otimizar a busca da amostra de menor distância, também serão necessárias todas as amostras para se fazer o cálculo de classificação. Enfim, o custo computacional para os três métodos (PID, SID e MID), da forma como foram expostos, é igual para todos, sendo da ordem de $O(N)$ e a memória requerida de mesma ordem $O(N)$. As técnicas de otimização do k-NN citadas em [BANDYOPADHYAY 2001], que poderiam ser usadas no método MID, certamente reduziriam o custo computacional, mas normalmente aumentam o requerimento de memória. Portanto, esses três algoritmos se tornariam inviáveis, quando o número de amostras fosse relativamente alto.

Um modo simples, intuitivo e muitas vezes funcional de resolver esse problema é utilizar o k-médias para reduzir N . Entretanto, será que o k-médias é capaz de reduzir significativamente o número de amostras sem perda de informação?

Analisando os métodos aqui estudados, há duas abordagens básicas de reconhecimento de padrões. A primeira, aparentemente mais intuitiva, é a de selecionar amostras e, de alguma forma, medir a proximidade das amostras com um padrão a ser classificado. Em outras palavras, o padrão a ser classificado é considerado um ponto no espaço de características que será classificado de acordo com as amostras mais próximas. Observa-se que as amostras também são consideradas pontos no espaço, cujas classificações são conhecidas. Geralmente, os métodos não paramétricos, tais como k-NN, k-médias, Parzen window, PID, SID e MID, utilizam essa abordagem. A outra abordagem divide o espaço de características (pela aprendizagem) em regiões, onde cada região corresponde a uma classe. O processo de classificação fica relativamente mais simples, pois é só verificar a que região do espaço de características pertence o padrão não classificado. O custo computacional da primeira abordagem tende a ser maior que o da segunda.

Pensou-se, então, em outra forma de representar padrões. Ao invés de usar amostras, que são pontos no espaço de características, várias amostras são substituídas por uma região no espaço. Há diversas maneiras de definir

regiões em um espaço, mas o ideal é definir uma região simples de representar matematica e computacionalmente, caso contrário, corre-se o risco de definir regiões cuja complexidade computacional é excessivamente maior do que os pontos envolvidos por ela, o que foge do objetivo de otimização. Portanto, tem-se que definir uma região de maneira simples e que envolva o maior número de amostras possível.

A maneira mais simples de definir uma região é mediante de um ponto e um raio, os quais no plano definem um círculo, no espaço 3D, uma esfera e, no espaço de D dimensões definem uma hiper-esfera (usando distância euclideana). No entanto, um círculo, quando envolve uma região relativamente grande, poderia distorcer a densidade de probabilidade, enquanto, definindo vários círculos, corre-se o risco de se ter quase tantas regiões quanto amostras (ver Figura 4.2).

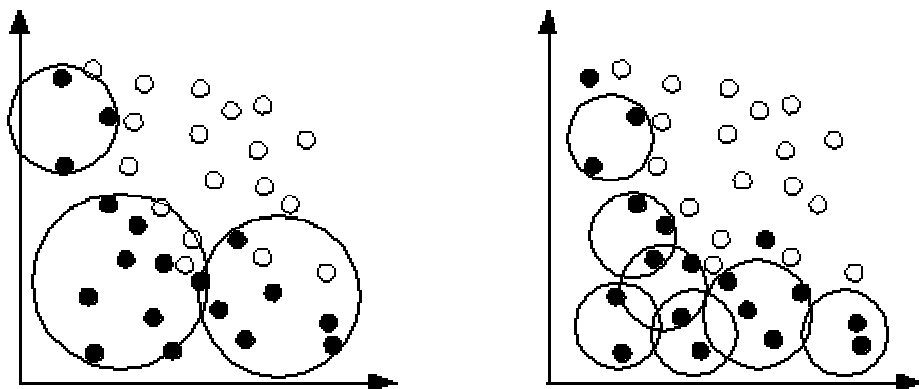


Figura 4.2 - Aglomeração circular

Representar padrões por regiões esféricas não é uma solução ruim, alguns métodos fazem isso razoavelmente. Analisando o equacionamento das redes RBF, nota-se que a função da camada intermediária é dividir o espaço de características em subespaços definidos por funções radiais, as quais definem regiões esféricas no espaço. Pode-se projetar uma rede RBF, usando o cálculo de distância Mahalanobis (que define regiões elipsoidais no espaço), o que minimizaria a distorção da densidade de probabilidade, mas aumentaria, significativamente, a complexidade computacional e o requerimento de memória devido às operações com a matriz de covariância.

A opção, então, foi definir regiões de padrões por de elipses ou elipsóides. Um elipsóide pode ser definido por dois pontos e um raio. Os dois pontos são os focos do elipsóide e o raio, que é constante, é soma das distâncias entre cada foco e a borda do elipsóide (alguns autores denominam essas distâncias de “raio A”, associado ao foco A, e “raio B” associado ao foco B) [SWOKOWISKI 1990]. Se os focos do elipsóide coincidirem no mesmo ponto, obtém-se uma esfera; se o raio do elipsóide for igual à distância entre os focos, obtém-se um segmento de reta e, se os focos coincidirem e o raio for nulo, obtém-se um ponto. Portanto, a mesma estrutura que representa um elipsóide pode representar um ponto, um segmento de reta ou uma esfera; logo, elipsóides podem representar, com boa precisão, regiões que definem padrões (Figura 4.3).

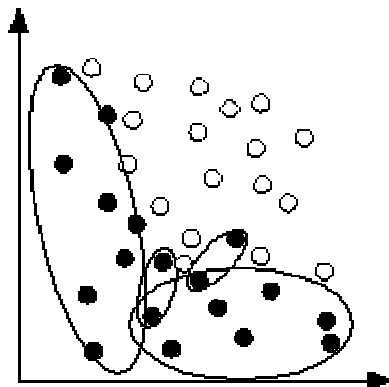


Figura 4.3 - Aglomeração elíptica

Um grupo de amostras pertencentes à mesma classe pode ser substituído por um elipsóide que as envolva, desde que esse elipsóide não envolva, também, amostras de outras classes. Da mesma forma que nos métodos PID, SID e MID, calculavam-se as distâncias entre o padrão a ser classificado e as amostras, calcula-se a distância entre o padrão a ser classificado e a elipsóide. Define-se, então, um elipsóide E pelos focos FA , FB e pelo raio R , sendo os focos vetores pertencentes ao espaço de características. Para que o elipsóide exista no espaço de características, o raio deve ser maior ou igual à distância entre os focos (equação 4.14). Desse modo, a distância de um ponto à borda do elipsóide pode ser calculada pela equação 4.15.

$$R_E \geq \text{dist}(FA, FB) \quad (4.14)$$

$$\text{distBordaE}(\mathbf{x}, E) = \text{dist}(\mathbf{x}, FA) + \text{dist}(\mathbf{x}, FB) - R_E \quad (4.15)$$

Note-se que, se a condição imposta pela equação 4.14 não for obedecida, não existirá um ponto \mathbf{x} para que $\text{distBordaE}(\mathbf{x}, E) = 0$; logo o elipsóide não existe para esse espaço. Caso $\text{distBordaE}(\mathbf{x}, E) < 0$, então, \mathbf{x} está dentro do elipsóide. Como o que interessa para reconhecimento de padrões é distância e não o quão dentro o ponto está na elipse, nesses casos, a distância será considerada zero. A distância de um ponto a um elipsóide é finalmente definida pela equação 4.16.

$$\text{distElip}(\mathbf{x}, E) = \begin{cases} \text{distBordaE}(\mathbf{x}, E); & \text{se } \text{distBordaE}(\mathbf{x}, E) > 0 \\ 0 & ; \text{se } \text{distBordaE}(\mathbf{x}, E) \leq 0 \end{cases} \quad (4.16)$$

Reformulando as equações de 4.9 a 4.13 e considerando a equação 4.16, é possível equacionar o PID elipsoidal (equações 4.17 e 4.18):

$$T_m = \sqrt[N]{\prod_{q=0}^Q \frac{1}{\text{distElip}(\mathbf{x}, E_q)} - \prod_{r=0}^R \frac{1}{\text{distElip}(\mathbf{x}, E_r)}} \quad (4.17)$$

$$T_m = \sqrt[Q]{\prod_{q=0}^Q \frac{1}{\text{distElip}(\mathbf{x}, E_q)} - \sqrt[R]{\prod_{r=0}^R \frac{1}{\text{distElip}(\mathbf{x}, E_r)}}} \quad (4.18)$$

SID elipsoidal (equações 4.19 e 4.20):

$$T_m = 2 \cdot \frac{\sum_{q=0}^Q \frac{1}{\text{distElip}(\mathbf{x}, E_q)} - \sum_{r=0}^R \frac{1}{\text{distElip}(\mathbf{x}, E_r)}}{N} \quad (4.19)$$

$$T_m = \frac{\sum_{q=0}^Q \frac{1}{\text{distElip}(\mathbf{x}, E_q)}}{Q} - \frac{\sum_{r=0}^R \frac{1}{\text{distElip}(\mathbf{x}, E_r)}}{R} \quad (4.20)$$

e MID elipsoidal (equação 4.21):

$$T_m = \frac{1}{\min(\text{distElip}(\mathbf{x}, E_q))} - \frac{1}{\min(\text{distElip}(\mathbf{x}, E_r))} \quad (4.21)$$

Para tornar as equações de 4.17 a 4.21 úteis, é preciso, primeiro, encontrar elipsóides que possam representar bem os padrões. Para isso foi desenvolvido um algoritmo de busca de elipsóides. O fluxograma da Figura 4.4

mostra os passos básicos do algoritmo desenvolvido e executado para cada dimensão de saída y , ou seja, M vezes. Os elipsóides são agrupados por classe, somente se for convencionalizado que cada dimensão de saída represente uma classe. Se cada classe for representada pela combinação das saídas, um único elipsóide pode representar parcialmente cada classe. No entanto, cada elipsóide poderá representar somente um estado de cada dimensão de saída. Para facilitar a explicação considera-se cada saída uma classe.

1. Entre todas as combinações de pares de amostra da mesma classe possíveis, procura-se o par mais distante entre si ainda não considerado (Figura 4.5a).
2. Verifica-se se as amostras encontradas não estão dentro de um elipsóide já formado, posto não ter sentido criar dois elipsóides um dentro do outro. Como o algoritmo é cíclico, é necessário marcar as amostras que já foram aglomeradas dentro de algum elipsóide.
3. Verifica-se se a elipse é válida, ou seja, se a região que ela vai ocupar pode realmente representar a classe. Essa verificação é feita, usando um dos três algoritmos PID, SID ou MID, optando-se pelo que tiver mais precisão na classificação para o caso. Deve-se escolher um limiar de semelhança, que convém ser entre 0,5 (meio) e 0,99 (noventa e nove centésimos). Em seguida, verifica-se se todo o segmento de reta definido pelo par de amostra pertence à classe. Se todo segmento estiver dentro do limiar de semelhança escolhido, o elipsóide será considerado válido; caso contrário, será descartado. Na Figura 4.5b é ilustrado um par de amostras descartado por não servir como elipsóide.

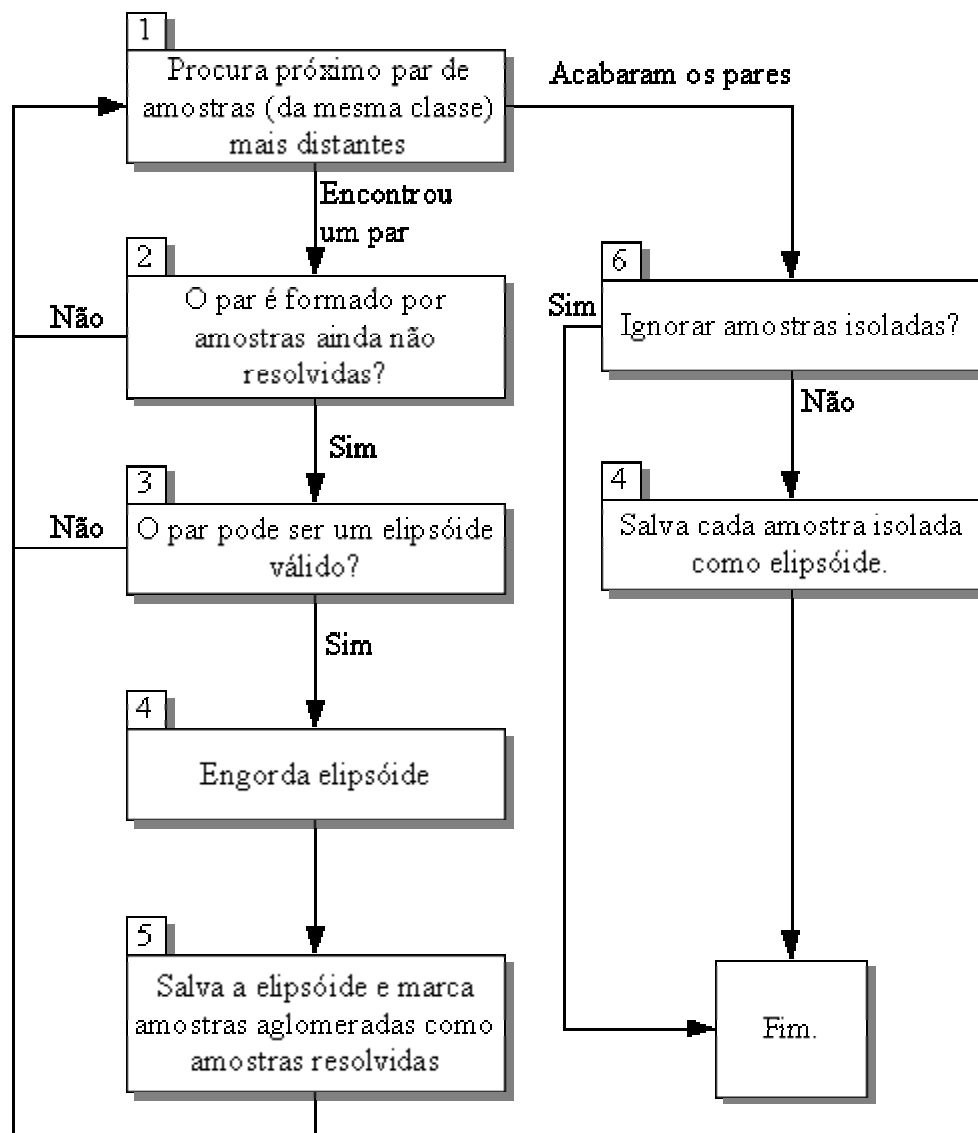


Figura 4.4 - Fluxograma do algoritmo de busca dos elipsóides

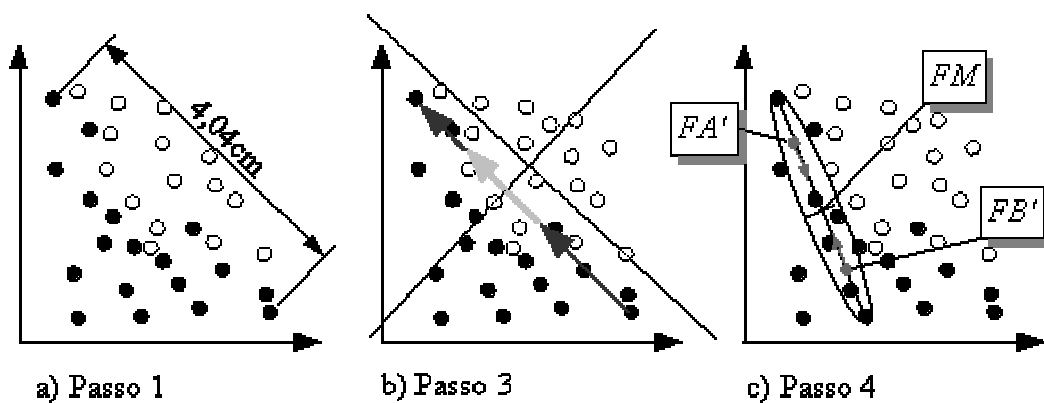


Figura 4.5 - Passos do fluxograma de busca dos elipsóides

4. Cria-se, então, um elipsóide, sendo que, a princípio, os focos provisórios FA' e FB' ficam sendo as próprias amostras e o raio R já é fixado na distância entre as duas amostras ($R=dist(FA, FB)$). Se os focos fossem mantidos como estão, esse elipsóide seria apenas um segmento de reta e não englobaria nenhuma outra amostra. Faz-se, então, com que os focos vão, passo a passo, caminhando simultaneamente para o ponto médio FM entre os focos FA' e FB' ($FM=(FA+FB)/2$). A cada passo verifica-se se alguma amostra pertencente a outra classe foi englobada pelo elipsóide. Caso isso ocorra, volta-se um passo e define-se o elipsóide; caso contrário, os focos vão tendendo ao centro do elipsóide (FM), o que faz com que o que antes era um segmento de reta passe a tender a uma hipersfera. Durante esse processo a tendência é o elipsóide englobar cada vez mais amostras de sua classe. A cada amostra englobada de mesma classe, FA' e FB' são respectivamente atribuídos a FA e FB definitivos. O processo pára, quando FA' e FB' chegam a FM ou quando se engloba uma amostra de outra classe (ver Figura 4.5c).
5. Todas as amostras englobadas no passo anterior devem ser marcadas como amostras resolvidas. Por fim, o elipsóide é armazenado.
6. Verifica-se se as amostras isoladas devem ser ignoradas.
7. Salva-se cada amostra isolada como elipsóide.

5 Softwares

Para aplicar as técnicas de reconhecimento de padrões, foram desenvolvidos dois programas para a comparação entre os métodos: classificador e mapeamento, ambos desenvolvidos em Delphi 6 [CANTÚ 1997][O'BRIEN 1996].

O compilador Delphi utiliza a linguagem de programação Pascal. Todas as rotinas de reconhecimento de padrões feitas para os softwares foram escritas em código Pascal “puro” (sem utilizar as funções específicas do Delphi) para possibilitar fácil portabilidade. Assim, as mesmas rotinas podem ser recompiladas para qualquer plataforma que possua um compilador de código Pascal. Além disso, as mesmas rotinas podem ser usadas por compiladores de C++ o próprio Delphi é capaz de compilar o código Pascal para bibliotecas em C++; e o GCC (Gnu Compiler Collection), que funciona em Linux, Windows e outros, também é capaz de compilar o código Pascal. Isso tudo garante uma grande portabilidade de todo código funcional dos métodos de reconhecimento de padrões, tendo apenas que se refazer a interface. Sendo assim, para fazer qualquer software de aplicação de reconhecimento de padrões, independente de plataforma, pode-se utilizar as mesmas rotinas sem a necessidade de reescrever os códigos, tendo, apenas, de fazer pouca ou nenhuma adaptação.

Nos softwares classificador e mapeamento foram aplicadas todas as rotinas necessárias para reconhecimento de padrões por curva normal, Parzen window, k-NN, perceptron multicamadas, PID, SID, MID e elipsoidal.

5.1 Classificador

O software *classificador* foi desenvolvido para comparar os métodos desenvolvidos aqui com técnicas tradicionais. Também pode ser usado como ferramenta de auxílio para o projeto de sistemas de reconhecimento de padrões. A entrada de dados para o software classificador pode ser um conjunto de amostras de qualquer tipo, desde que cada amostra já esteja no formato de vetor de característica, ou seja, os dados já devem ter passado pelo

bloco de extração de características. O software classificador faz o papel do bloco classificador de uma maneira bastante flexível (Figura 2.3, pág. 6). A princípio, desde que extraídas adequadamente as características, o software deve ser capaz de classificar qualquer tipo de dado.

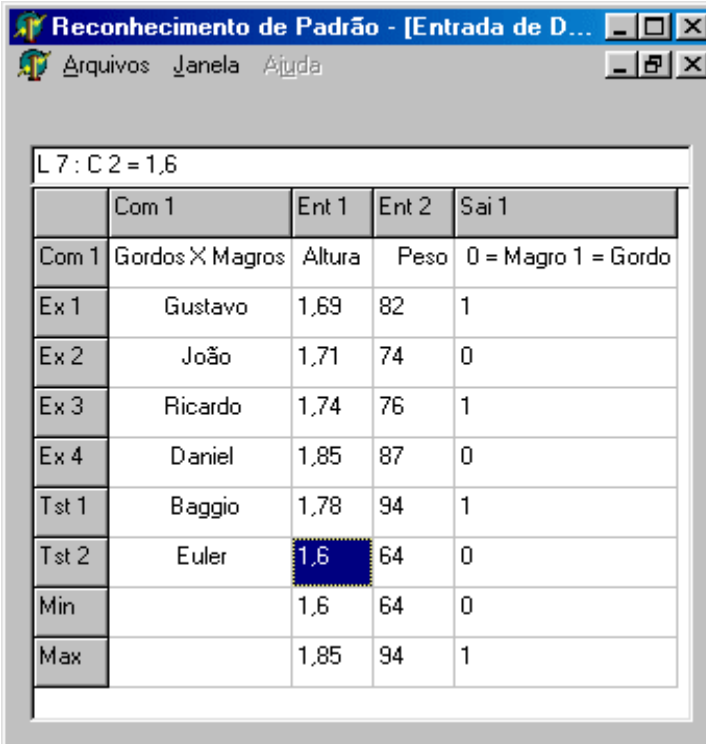
Os dados de entrada podem ser importados por tabelas no formato CSV (comma separate value - valor separado por vírgula). No formato de arquivo CSV, as colunas de dados são separadas por ponto-e-vírgulas e cada linha de dados termina com um caracter de mudança de linha. Cada linha representa uma amostra, cada coluna é uma dimensão do vetor de entrada \mathbf{x} ou do vetor de saída \mathbf{y} . Para exemplificar será usado o exemplo da classificação de pessoas magras e obesas. Abaixo, Figura 5.1, um exemplo fictício ilustra como deve ser o formato de um arquivo CSV que pode ser editado em qualquer editor de texto (Notepad, por exemplo). Normalmente, as planilhas eletrônicas também são capazes de salvar em arquivos CSV.

Gordos X Magros;	Altura;	Peso;	0 = Magro 1 = Gordo
Gustavo;	1,69;	82;	1
João;	1,71;	74;	0
Ricardo;	1,74;	86;	1
Daniel;	1,85;	87;	0
Baggio;	1,78;	96;	1
Euler;	1,60;	64;	0

Figura 5.1 - Exemplo do formato CSV

Quando se abre um arquivo de dados pela primeira vez, é necessário marcar o significado de cada linha e coluna (Figura 5.2). As linhas podem ser marcadas como comentário (indicadas pelo prefixo “Com”), amostras de exemplo (por “Ex”) ou amostras de teste (por “Tst”); as colunas podem ser marcadas como comentário (também indicadas por “Com”), dimensão do vetor de entrada (indicada por “Ent”) ou dimensão do vetor de saída (indicada por “Sai”). As amostras marcadas como teste não participam do treinamento de nenhum método e serão aplicadas aos métodos de reconhecimento para

verificar se a classificação está sendo satisfatória. O software permite fazer a normalização dos dados, ou seja, aplicar um fator de escala a cada dimensão dos vetores a fim de normalizá-los entre 0 e 1, mas antes deve-se calcular os valores máximos de cada dimensão (indicados por “Min”, valor mínimo, e “Max”, valor máximo). Teoricamente, o número máximo de linhas e colunas é 65.535 (sessenta e cinco mil quinhentos e trinta e cinco) mas, na prática, esse número é limitado pela configuração do computador.



L 7: C 2 = 1,6

	Com 1	Ent 1	Ent 2	Sai 1
Com 1	Gordos X Magros	Altura	Peso	0 = Magro 1 = Gordo
Ex 1	Gustavo	1,69	82	1
Ex 2	João	1,71	74	0
Ex 3	Ricardo	1,74	76	1
Ex 4	Daniel	1,85	87	0
Tst 1	Baggio	1,78	94	1
Tst 2	Euler	1,6	64	0
Min		1,6	64	0
Max		1,85	94	1

Figura 5.2 - Tela de entrada de dados do software classificador

Muitas vezes, a quantidade de amostras disponível não é suficiente para separar um número razoável de amostras de teste ou, mesmo, para garantir que o treinamento está sendo adequado. Por isso, foi criada a opção de gerar amostras pseudo-aleatórias com base nas amostras existentes (Figura 5.3). Escolhendo um desvio-padrão e quantas amostras serão criadas por amostras existentes, o software gera novas amostras, obedecendo à distribuição normal e considerando as amostras originais como médias. Desse modo, para efeito de pesquisa, é possível criar um grande número de amostras com uma distribuição lógica conhecida e com qualquer grau de liberdade entre

os padrões.

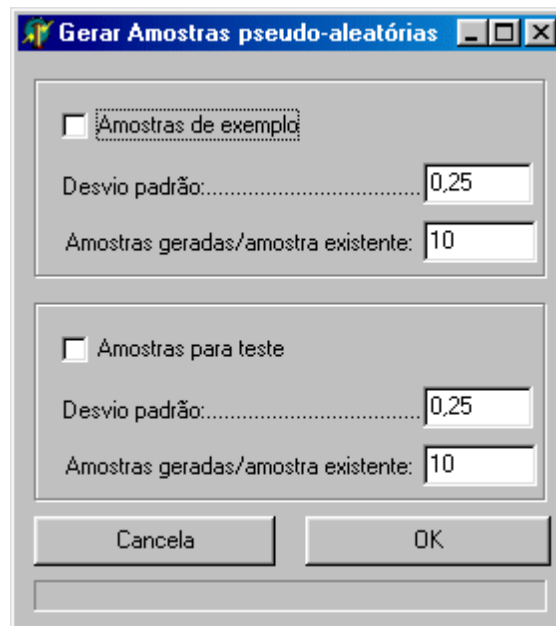


Figura 5.3 - Tela para gerar amostras pseudo-aleatórias no software classificador.

Depois de definir toda a tabela de amostras, os métodos de reconhecimento de padrões serão testados. Na janela “método” do software classificador (Figura 5.4).

Na palheta “geral” da janela de seleção de método são exibidas as dimensões do problema de reconhecimento, possibilitando fazer um teste automático dos métodos PID, SID, MID e elipsoidal, usando-se a técnica de deixar uma amostra de fora. Essa técnica é utilizada, quando não há amostras de teste suficientes para a validação do método de reconhecimento e consiste em deixar de fora uma amostra de exemplo para ser usada como amostra de teste, repetindo-se o mesmo processo para todas as amostras. Desse modo, todas as amostras disponíveis são usadas para teste e treinamento sem necessidade de gerar amostras aleatórias.

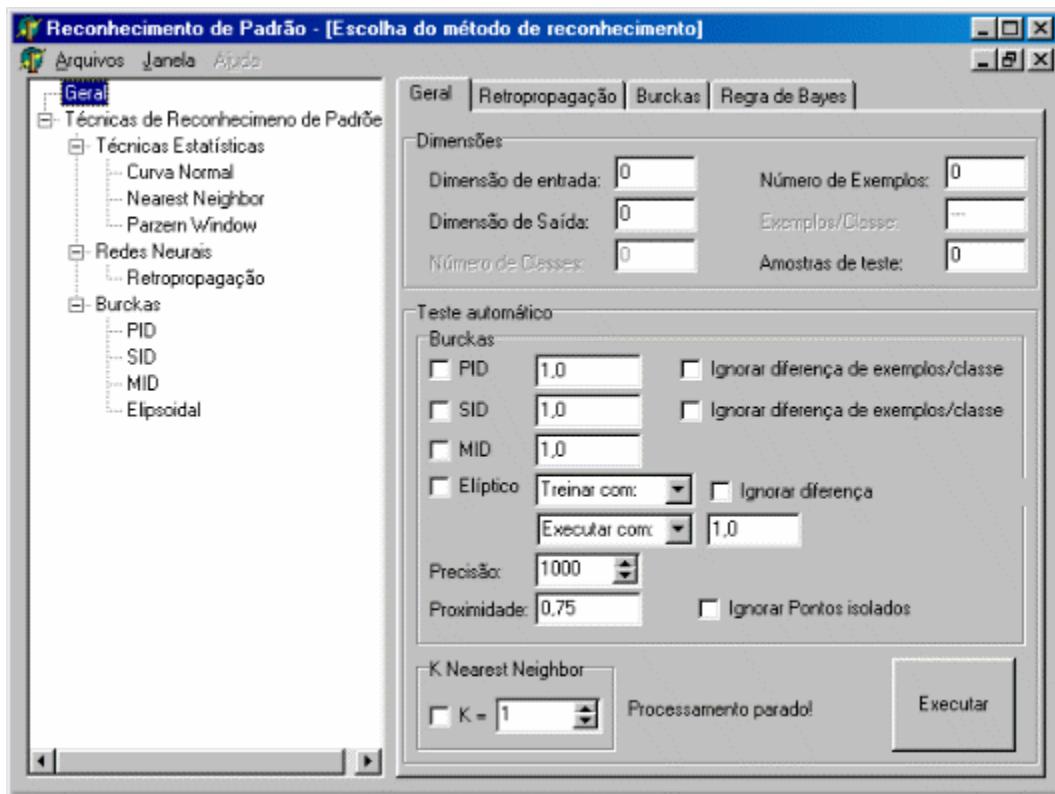


Figura 5.4 - Janela de escolha dos métodos de reconhecimento de padrões

Na palheta “retropropagação” (Figura 5.5) é feito o treinamento de rede perceptron multicamadas, usando-se algoritmo de retropropagação. O treinamento é feito da forma mais simples, sem a constante de momento e observando apenas o erro das amostras de exemplo, não as amostras de teste. Após ter sido realizado o treinamento, ao pressionar o botão “executar”, a rede será testada com cada amostra de teste e os resultados, exibidos na janela de saída.

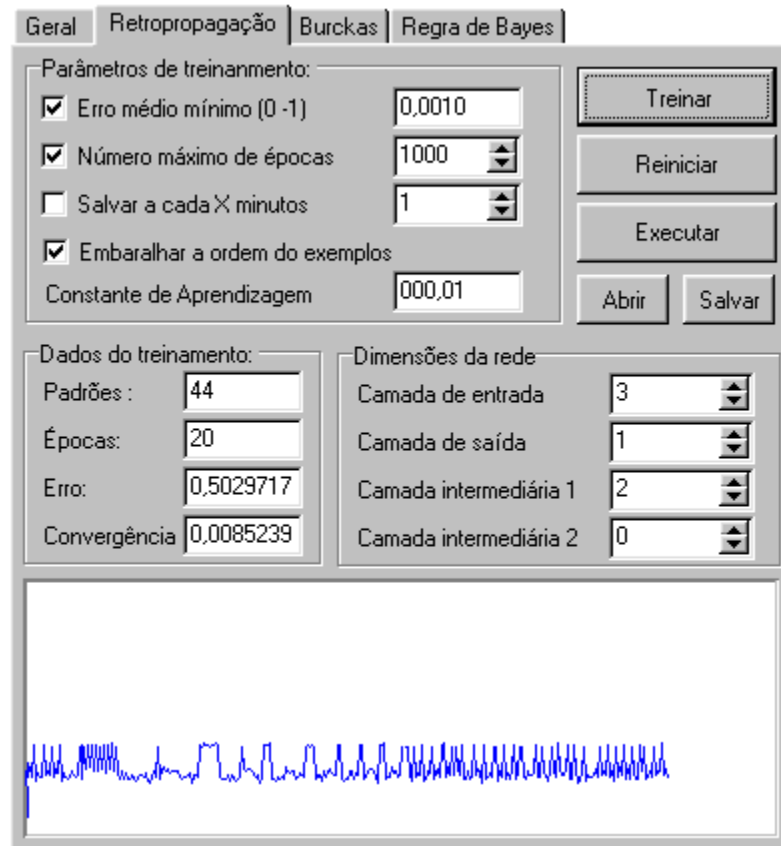


Figura 5.5 - Treinamento de rede perceptron multi-camadas

Na palheta “Regra de Bayes” (Figura 5.6) são escolhidas as técnicas de reconhecimento estatístico e ajustados os respectivos parâmetros: curva normal, Parzen window, e k-NN. Do mesmo modo, quando pressionado o botão “executar”, as amostras de teste são aplicadas a cada uma das técnicas selecionadas e os resultados, exibidos na janela de saída.



Figura 5.6 - Palheta “Regra de Bayes”

Na palheta “Burckas” (Figura 5.7) selecionam-se os métodos PID, MID, SID e elipsoidal. Deve-se ajustar os respectivos graus de generalização e parâmetros para o algoritmo de busca dos elipsóides. Há um botão “executar” separado para o método elipsoidal pois, normalmente, a busca dos elipsóides deve ser feita depois de se verificar quais dos três métodos (PID, SID ou MID) tem o melhor desempenho. Esse será o método utilizado no algoritmo de busca.

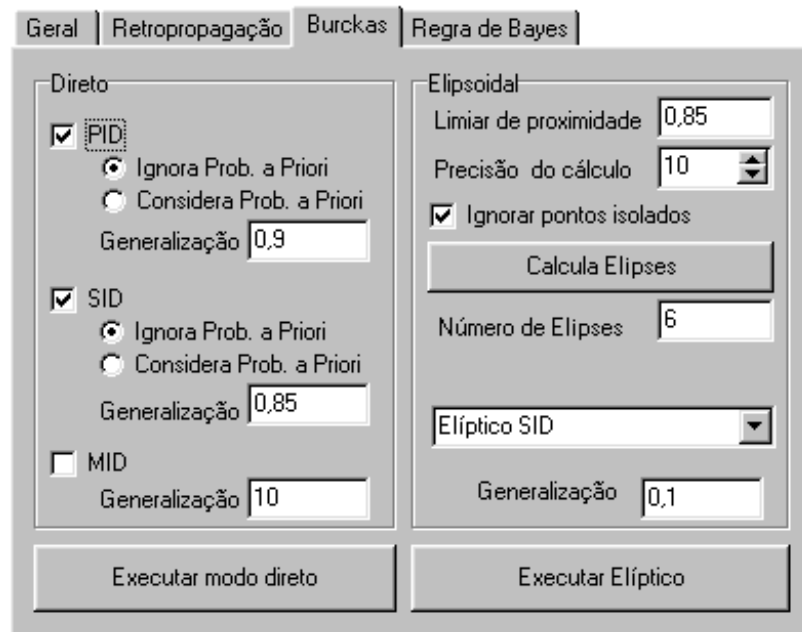


Figura 5.7 - Palheta de seleção dos métodos PID, SID, MID e elipsoidal

Na janela de saída de dados os resultados são colocados em forma de tabela. Antes de cada seqüência de teste é indicado o nome do método usado, sendo que, cada linha é um teste. Cada duas colunas referem-se a uma dimensão de saída, sendo a primeira a resposta obtida pelo método e a segunda, a porcentagem da proximidade com a resposta considerada correta pela amostra de teste.

A tabela da janela de saída pode ser salva em arquivo no formato CSV e exportada para uma planilha eletrônica, onde é possível fazer uma análise mais detalhada dos dados.

5.2 Mapeamento

O software “mapeamento” foi desenvolvido para visualizar como cada método de reconhecimento define a fronteira de decisão. No entanto, só é

possível mapear problemas de reconhecimento cujo vetor de entrada é de duas dimensões e o vetor de saída de apenas uma dimensão. A saída do programa mapeamento é um gráfico de superfície, onde o vetor de entrada é representado pelas coordenadas xy e a saída, pela cor no ponto. A Figura 5.8 mostra um gráfico de superfície (Figura 5.8a) representado por um gráfico de cores (Figura 5.8b) da mesma maneira que é mostrado no programa mapeamento.

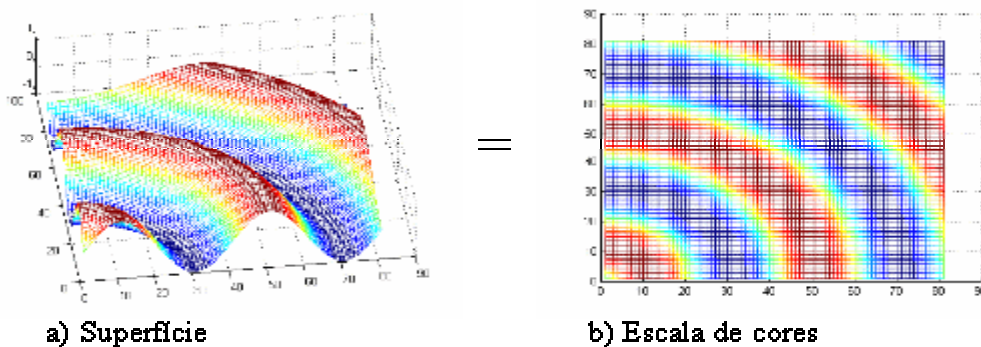


Figura 5.8 - Gráfico de superfície = gráfico de cores

O software mapeamento permite que as amostras sejam criadas manualmente, simplesmente selecionando a classe desejada e, em seguida, clicando sobre a área do mapeamento. A Figura 5.9 exibe parte da tela do software mapeamento exibindo o mapa de resposta de uma rede perceptron multicamadas. As amostras de treinamento são os círculos, sendo cada pixel, no mapa, considerado uma amostra de teste. A saída é em tons de cinza, de acordo com a escala à direita do gráfico.

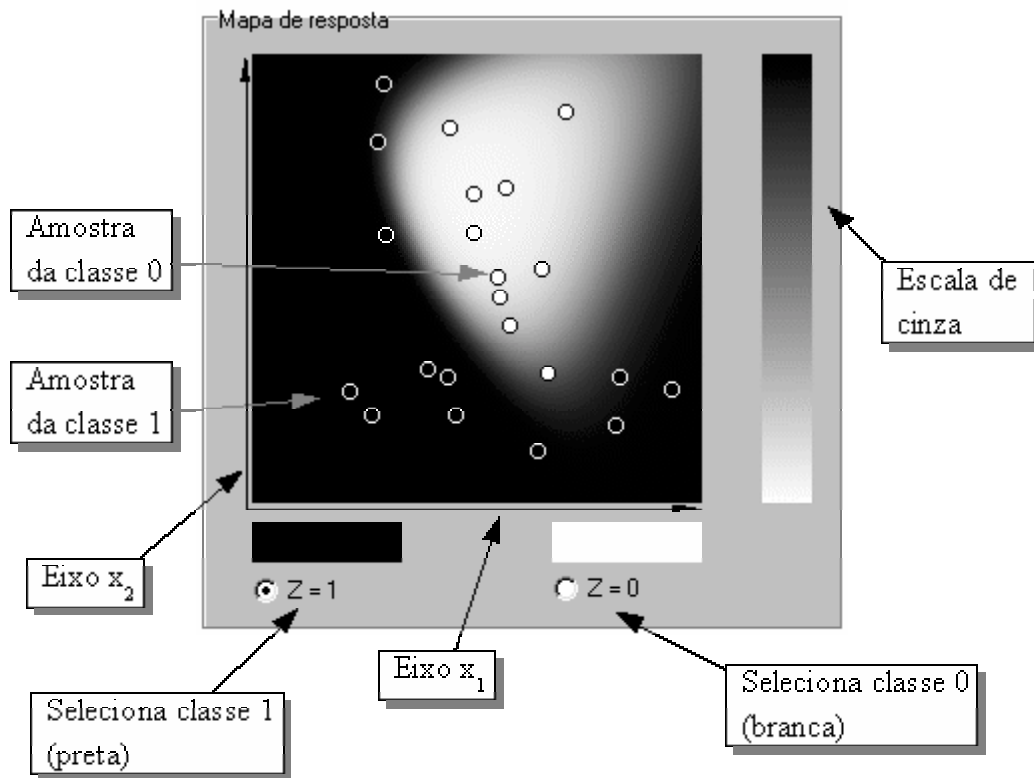


Figura 5.9 - Programa Mapeamento

6 Estudo Comparativo

Para fazer um estudo comparativo entre os diferentes métodos de reconhecimento de padrões, optou-se por gerar os conjuntos de amostras, o que torna possível controlar o seu número, o grau de liberdade entre as amostras e o modo como as classes estão distribuídas no espaço de características. Desse modo é possível fazer uma comparação quantitativa, verificando qual a taxa de acerto de cada método em diferentes distribuições de classes.

Para criar amostras que tivessem uma distribuição não linear foi criado, primeiro, um conjunto de amostras representando a lógica do “ou exclusivo” (“OU exclusivo (XOR)”,- conforme a Tabela 6.1). Para que as amostras não ficassem nos extremos do espaço vetorial, ao invés de 0, foi utilizado 0,33 e, ao invés de 1, 0,66.

Tabela 6.1 - OU exclusivo

x_1	x_2	y
0,33	0,33	0
0,66	0,66	0
0,33	0,66	1
0,66	0,33	1

A Figura 6.1 mostra a distribuição das amostras primárias do “ou exclusivo”. Em cima de cada amostra primária do “ou exclusivo” foram geradas as outras amostras.

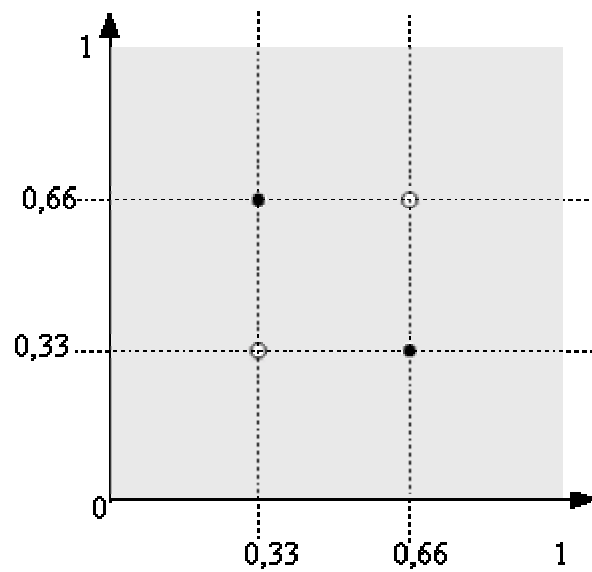


Figura 6.1 - OU exclusivo primário.

Cada amostra é considerada como um ponto médio, escolhe-se, então, um desvio-padrão e quantas amostras serão derivadas das amostras primárias. Assim, as amostras geradas obedecem a uma distribuição normal ao redor de cada amostra primária. Quanto maior for o desvio-padrão maior será o grau de liberdade das amostras, ou seja, muito mais complicado para classificar. Por exemplo, para o teste 3 (a ser discutido posteriormente) foram criadas 100 amostras para cada amostra primária com desvio-padrão de 0,1 (Figura 6.2).

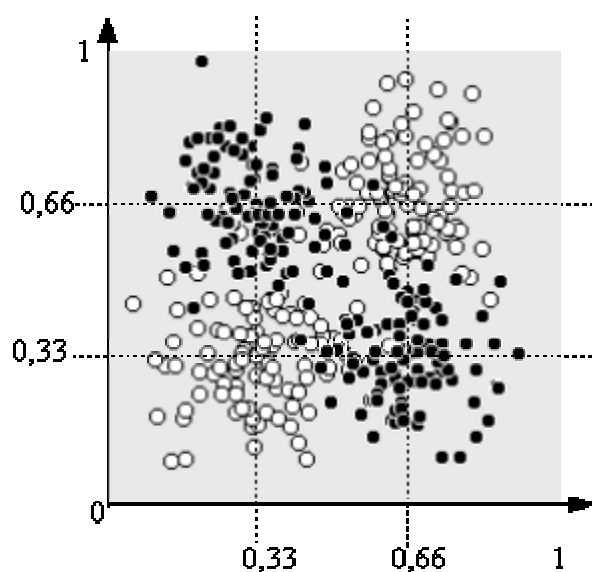


Figura 6.2 - OU exclusivo, Teste 3.

Da mesma maneira, foram criados nove conjuntos de testes (conforme Tabelas de 6.2 a 6.10), variando-se o número de amostras de exemplo e o desvio-padrão. Para todos os testes foram criadas 404 amostras de teste com o mesmo grau de dispersão das amostras de exemplo.

Os nove grupos de teste foram submetidos aos seguintes métodos de classificação: PID, SID, MID, elipsoidal, curva normal, k-NN, Parzen window e rede neural perceptron multicamadas. Utilizou-se $k = \sqrt{N}$ para ajustar o valor de k para o k-NN e $h^2 = 1/\sqrt{N}$ para a largura da janela em Parzen window [DUDA 2001]. O algoritmo de busca das elipses foi configurado para usar o PID verificando as elipses válidas com um limiar de 0,85 de proximidade; a precisão de cálculo* foi de 30 (quanto maior mais preciso) e as amostras isoladas foram ignoradas. O método elipsoidal foi executado utilizando o PID elipsoidal (equação 4.18). As redes perceptron foram treinadas sem utilizar momento, o critério de parada foi a observação do erro médio do conjunto de treinamento e a configuração da rede e o erro médio mínimo foram diferentes para cada teste, pois procurou-se uma configuração que possibilitasse o

* A precisão do cálculo utilizada nos passos 3 e 4 é descrita no fluxograma do algoritmo de busca dos elipsóides (Figura 4.4 na página 42).

treinamento. Os resultados obtidos e os detalhes de cada teste estão nas Tabelas de 6.2 a 6.10 e nos Gráficos de 6.1 a 6.9. A saída y de todos os métodos é entre 0 e 1; se $y < 0,5$, o padrão é classificado como 0; se $y > 0,5$ o padrão é classificado como 1 e, se $y=0,5$, o padrão é considerado incerto. Foi considerado um acerto toda vez que a classe calculada pelo método correspondia à amostra de teste. Nos casos em que o padrão era considerado incerto foi considerado meio acerto. Todos os testes foram realizados com o software classificador.

Tabela 6.2 - Teste 1, realizado com amostras com baixo desvio-padrão e poucas amostras de exemplo. O número de amostras de teste foi 404 para todos os testes, independentemente do número de amostras de exemplo.

Teste 1		
Nº de amostras:	44	
Desvio Padrão:	0.1	
	Acerto	Percentual
PID	366	90.59%
SID	357	88.37%
MID	345	85.40%
Elipsoidal	324	80.20%
Normal	362.5	89.73%
k-NN	362	89.60%
Parzen W.	346.5	85.77%
RNA	361	89.36%

Elipsoidal - Nº de Elipsóides:	9
k-NN - k:	7
Parzen W. - h:	0.38
RNA 2-10-1 - Erro <	0.1

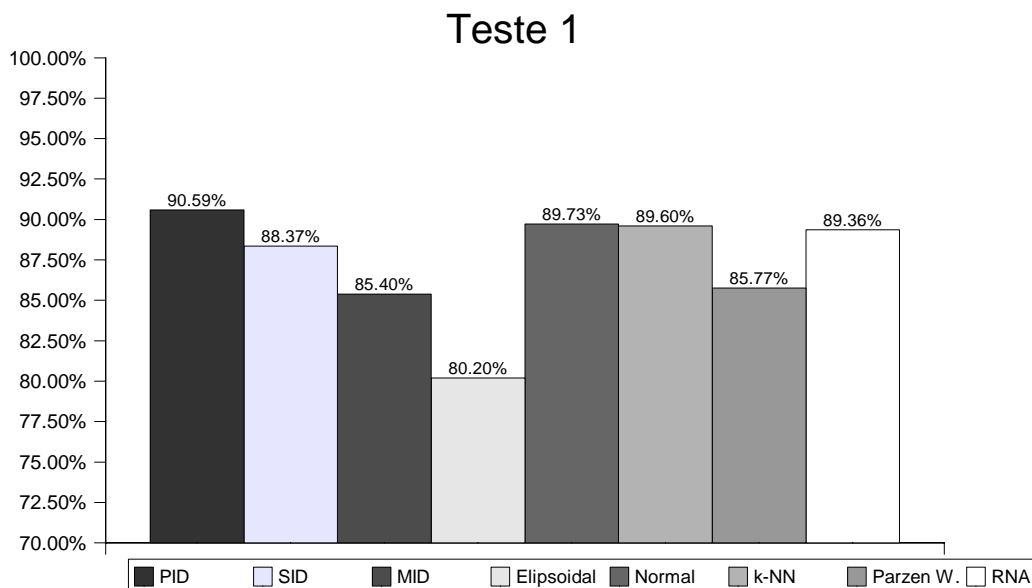


Gráfico 6.1 - Percentual de acerto do teste 1 - Percentual de acerto de cada método. Possivelmente por causa do baixo número de amostras, os métodos MID e o elipsoidal não tiveram um desempenho tão bom.

Tabela 6.3 - Teste 2, realizado com amostras com baixo desvio-padrão e médio número de amostras de exemplo. O número de amostras de teste foi mantido em 404.

Teste 2		
Nº de amostras:	204	
Desvio Padrão:	0.1	
	Acertos	Percentual
PID	366	90.59%
SID	357	88.37%
MID	345	85.40%
Elipsoidal	371	91.83%
Normal	362.5	89.73%
k-NN	363	89.85%
Parzen W.	365	90.35%
RNA	361	89.36%

Elipsoidal - Nº de Elipsóides:	23
k-NN - k:	14
Parzen W. - h:	0.27
RNA 2-10-1 - Erro <	0.15

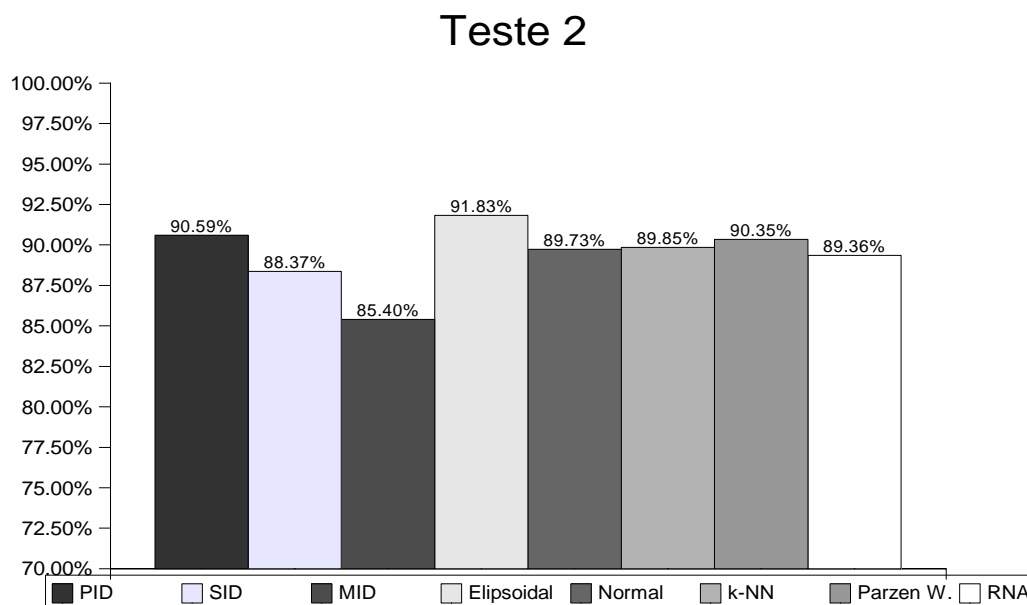


Gráfico 6.2 - Percentual de acerto do teste 2. Com o aumento do número de amostras de teste, pode-se notar que, em média, o percentual de acerto aumentou. Nesse teste destaca-se o método elipsoidal que no teste 1 teve baixo desempenho, já aqui teve melhor desempenho.

Tabela 6.4 - Teste 3, realizado com amostras com baixo desvio-padrão e um grande número de amostras de exemplo.

Teste 3		
Nº de amostras:	404	
Desvio Padrão:	0.1	
	Acertos	Percentual
PID	364	90.10%
SID	361	89.36%
MID	349	86.39%
Elipsoidal	360	89.11%
Normal	364	90.10%
k-NN	359.5	88.99%
Parzen W.	360	89.11%
RNA	356	88.12%

Elipsoidal - Nº de Elipsóides:	43
k-NN - k:	20
Parzen W. - h:	0.05
RNA 2-10-1 - Erro <	0.22

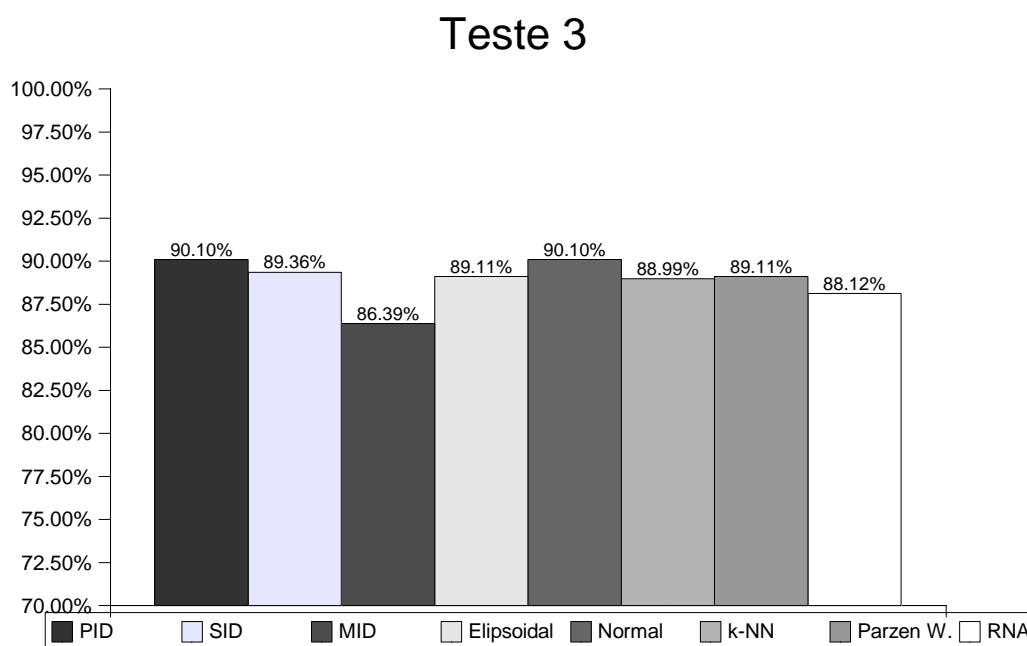


Gráfico 6.3 - Percentual de acerto do teste 3. O percentual de acerto nesse teste foi mais bem equalizado em relação aos testes anteriores.

Tabela 6.5 - Teste 4, realizado com amostras com médio desvio-padrão e poucas amostras de exemplo.

Teste 4		
Nº de amostras:	44	
Desvio Padrão:	0.15	
	Acertos	Percentual
PID	277	68.56%
SID	275	68.07%
MID	265	65.59%
Elipsoidal	271	67.08%
Normal	285.5	70.67%
k-NN	280	69.31%
Parzen W.	294.5	72.90%
RNA	281	69.55%

Elipsoidal - Nº de Elipsóides:	10
k-NN - k:	7
Parzen W. - h:	0.38
RNA 2-10-1 - Erro <	0.15

Teste 4

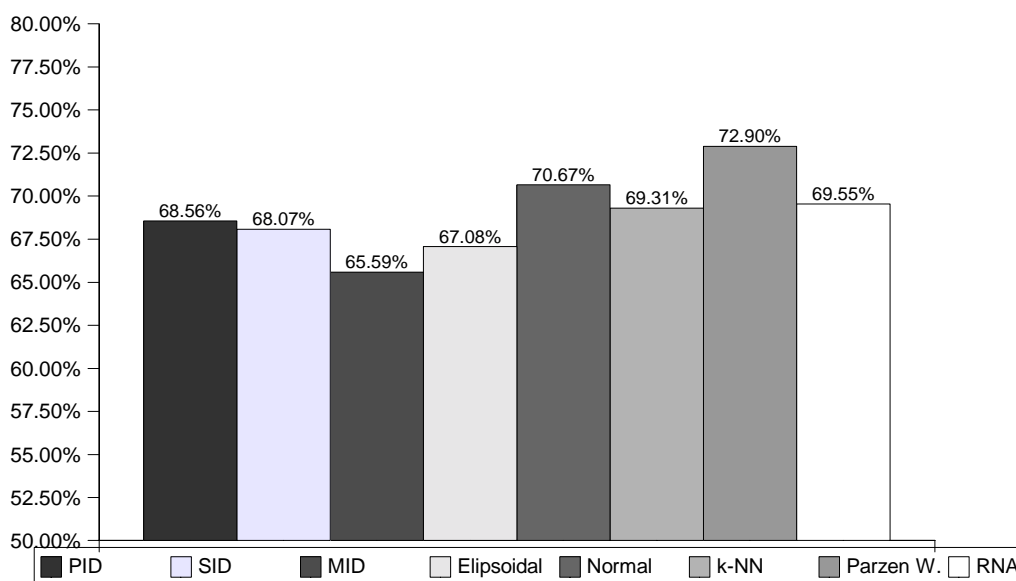


Gráfico 6.4 - Percentual de acerto do teste 4. Com o aumento do desvio-padrão, a média do percentual de acerto diminui. Neste caso, há um destaque para o método Parzen window que obteve o melhor desempenho.

Tabela 6.6 - Teste 5, realizado com amostras com médio desvio-padrão e número de amostras de exemplo.

Teste 5		
Nº de amostras:	204	
Desvio Padrão:	0.15	
	Acerto	Percentual
PID	292	72.28%
SID	274	67.82%
MID	252	62.38%
Elipsoidal	278.5	68.94%
Normal	306.5	75.87%
k-NN	301.5	74.63%
Parzen W.	297.5	73.64%
RNA	299	74.01%

Elipsoidal - Nº de Elipsóides:	36
k-NN - k:	14
Parzen W. - h:	0.27
RNA 2-10-1 - Erro <	0.3

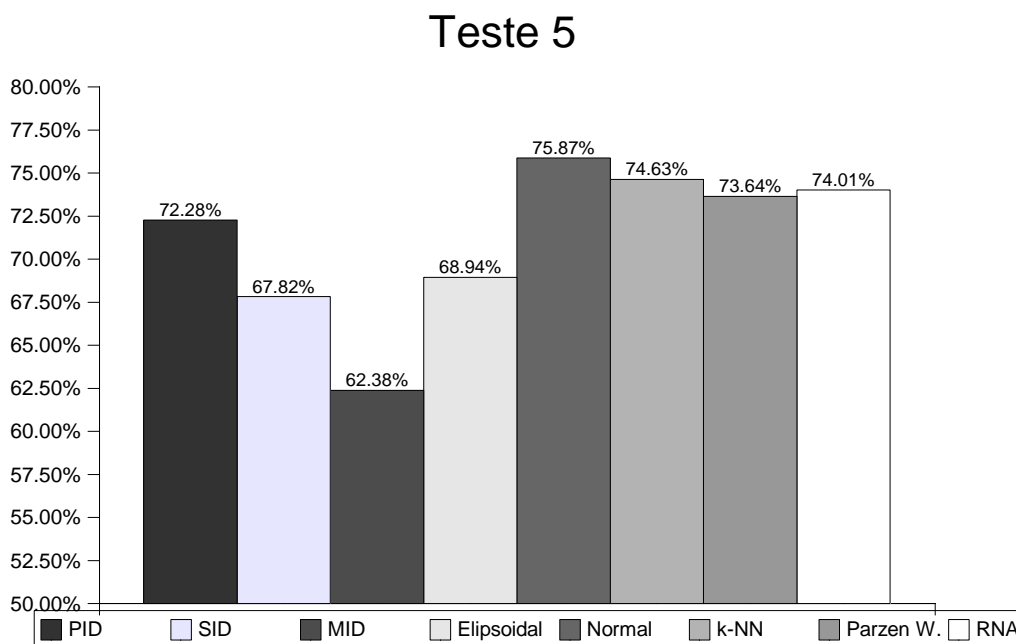


Gráfico 6.5 - Percentual de acerto do teste 5, que teve em média um percentual maior que o teste 4 simplesmente pelo aumento do número de amostras de exemplo.

Tabela 6.7 - Teste 6, realizado com amostras com médio desvio-padrão e muitas amostras de exemplo.

Teste 6		
Nº de amostras:	404	
Desvio Padrão:	0.15	
	Acerto	Percentual
PID	302	74.75%
SID	288	71.29%
MID	272	67.33%
Elipsoidal	295	73.02%
Normal	298	73.76%
k-NN	300.5	74.38%
Parzen W.	294.5	72.90%
RNA	221	54.70%

Elipsoidal - Nº de Elipsóides:	103
k-NN - k:	20
Parzen W. - h:	0.22
RNA 2-30-20-1 - Erro <	0.15

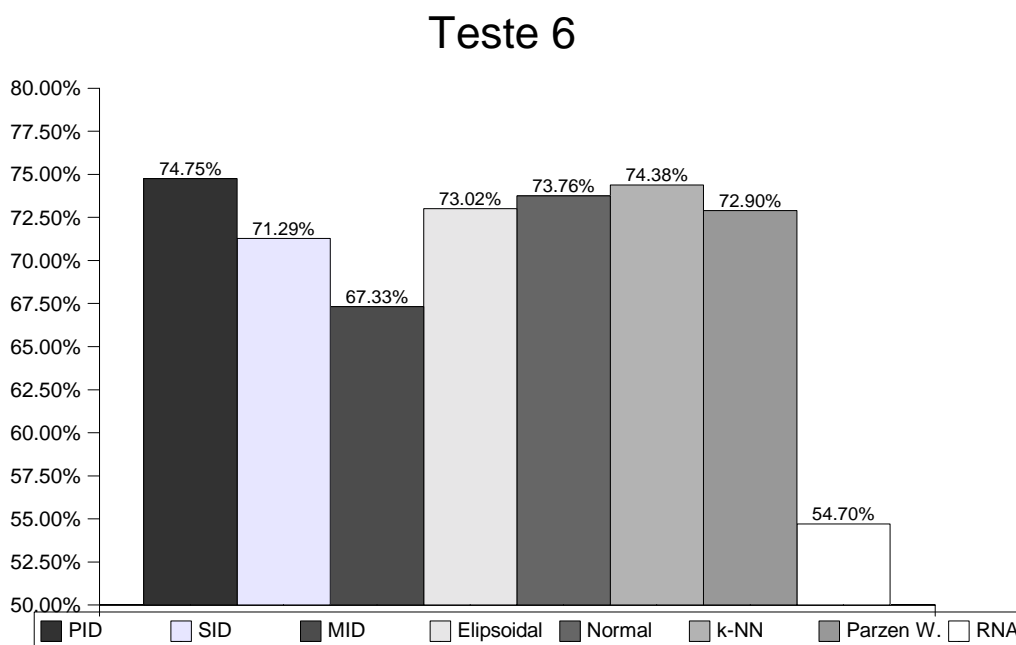


Gráfico 6.6 - Percentual de acerto do teste 6. Com um número ainda maior de amostras a maioria dos métodos apresentou uma melhora no percentual de acerto, entretanto, foi difícil encontrar uma boa configuração para treinar a rede neural.

Tabela 6.8 - Teste 7, realizado com amostras com alto desvio-padrão e poucas amostras de exemplo.

Teste 7		
Nº de amostras:	44	
Desvio Padrão:	0.2	
	Acerto	Percentual
PID	253	62.62%
SID	254.5	63.00%
MID	248	61.39%
Elipsoidal	263	65.10%
Normal	271	67.08%
k-NN	273	67.57%
Parzen W.	261.5	64.73%
RNA	249.5	61.76%

Elipsoidal - Nº de Elipsóides:	13
k-NN - k:	7
Parzen W. - h:	0.38
RNA 2-20-20-1 - Erro <	0.2

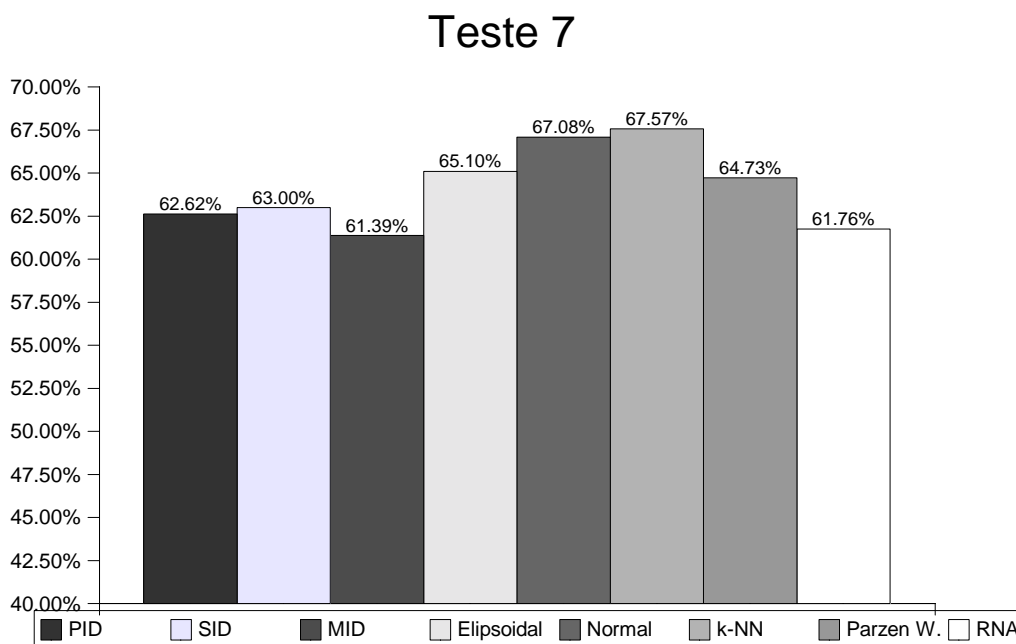


Gráfico 6.7 - Percentual de acerto do teste 7. Esse é o teste que obteve a menor média de percentual de acerto, em razão do ao alto desvio-padrão e baixo número de amostras.

Tabela 6.9 - Teste 8, realizado com amostras com alto desvio-padrão e um número médio de amostras de exemplo.

Teste 8		
Nº de amostras:	204	
Desvio Padrão:	0.2	
	Acerto	Percentual
PID	248	61.39%
SID	238	58.91%
MID	238	58.91%
Elipsoidal	243	60.15%
Normal	269.5	66.71%
k-NN	256.5	63.49%
Parzen W.	250	61.88%
RNA	233.5	57.80%

Elipsoidal - Nº de Elipsóides:	39
k-NN - k:	14
Parzen W. - h:	0.27
RNA 2-30-20-1 - Erro <	0.3

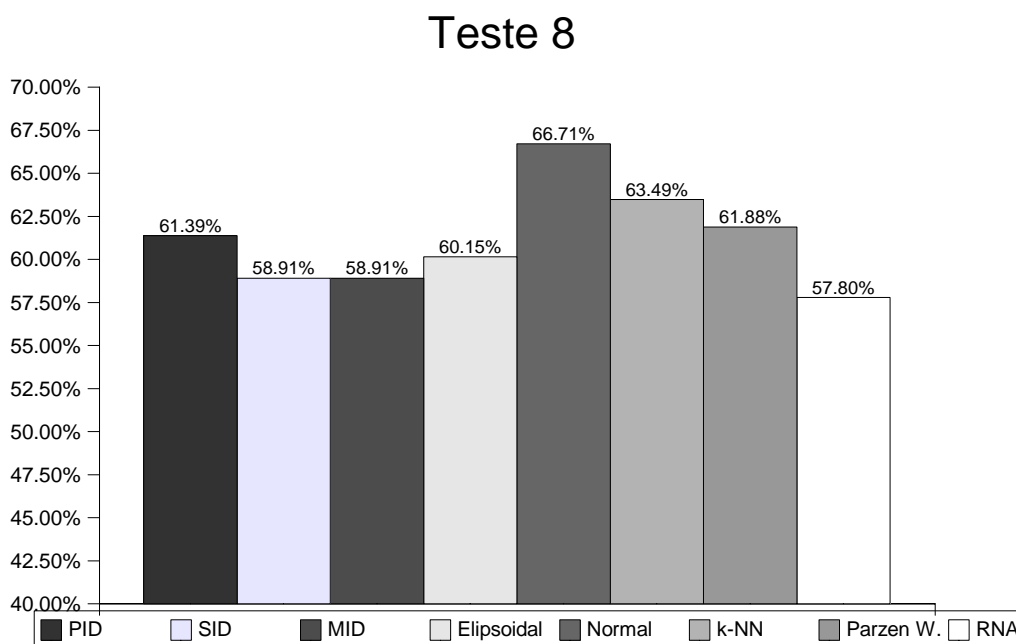


Gráfico 6.8 - Percentual de acerto do teste 8. Nesse teste o método pela curva normal obteve um resultado muito superior em relação aos outros.

Tabela 6.10 - Teste 9, realizado com amostras com alto desvio-padrão e um número alto de amostras de exemplo.

Teste 9		
Nº de amostras:	404	
Desvio Padrão:	0.2	
	Acerto	Percentual
PID	266.5	65.97%
SID	257.5	63.74%
MID	228	56.44%
Elipsoidal	264.5	65.47%
Normal	276	68.32%
k-NN	264.5	65.47%
Parzen W.	265.5	65.72%
RNA	202	50.00%

Elipsoidal - Nº de Elipsóides:	96
k-NN - k:	20
Parzen W. - h:	0.22
RNA 2-30-20-1 - Erro <	0.3

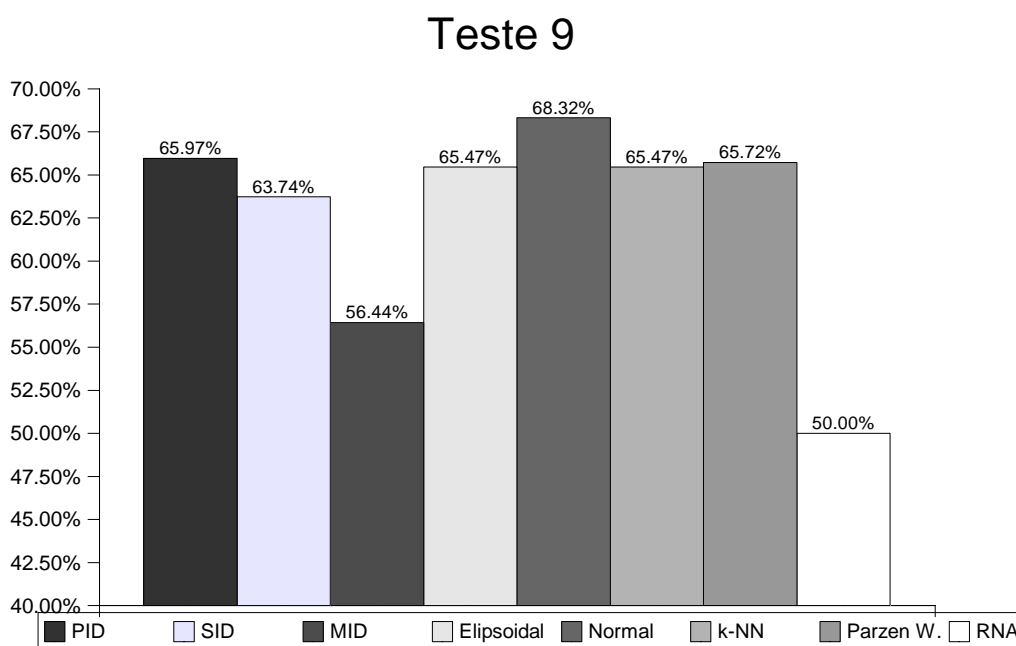


Gráfico 6.9 - Percentual de acerto do teste 9. Por causa de um grande número de amostras com o desvio-padrão elevado, foi muito complicado encontrar uma configuração para a rede neural que treinasse; em consequência, o percentual de acerto da rede caiu muito.

Os mapas de resposta de todos os métodos do teste 2 gerados pelo software mapeamento estão na Figura 6.3.

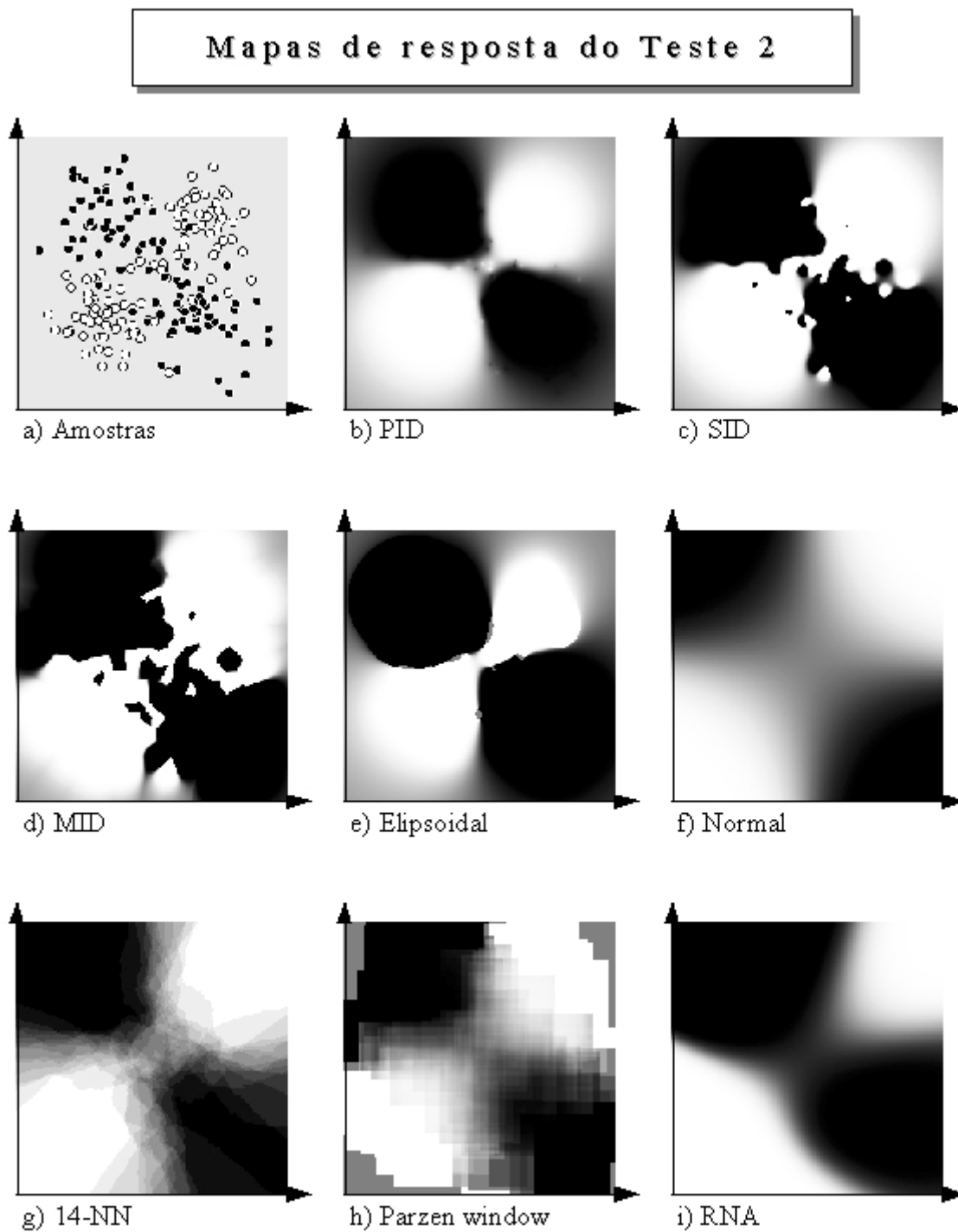


Figura 6.3 - Mapas de resposta do teste 2.

De maneira geral, pode-se dizer que todos os métodos tiveram um desempenho satisfatório. No entanto, nos testes 6, 8 e 9, o desempenho da rede neural foi relativamente baixo. Isso ocorreu porque nesses casos houve uma

dificuldade maior para encontrar uma configuração em que a rede convergisse para a aprendizagem que não significa ser impossível encontrar uma outra configuração para a rede que pudesse ter um desempenho melhor, mas mostra uma dificuldade no projeto de sistemas de reconhecimento de padrões usando redes neurais perceptron multicamadas.

Os métodos k-NN e Parzen window obtiveram, em média, os melhores desempenhos. Já era de esperar que os resultados fossem parecidos entre esses dois métodos, já que são quase equivalentes. Uma das diferenças entre ambos fica bem evidente, quando se comparam os mapas de respostas (Figuras 6.3g e 6.3h). Note-se que no Parzen window as regiões pertencentes à mesma probabilidade *a posteriori* são definidas por ângulos retos, já o k-NN possui muitas curvas. Esse efeito ocorreu porque o k-NN foi aplicado com distância euclideana, enquanto a janela do parzem window é equivalente à distância Manhattan. Outra diferença interessante é que a classificação de padrões que estão relativamente longe de qualquer amostra é sempre duvidosa pelo Parzen window. Já no k-NN, a classificação sempre tenderá à classe das amostras mais próximas. Esse fenômeno também pode ser observado no mapa de resposta, comparando-se o canto inferior esquerdo dos mapas das Figuras 6.3g e 6.3h, onde o k-NN classifica como a classe "0" (branco) e o Parzen window classifica como duvidoso (cinza). É importante salientar que os critérios (citados antes) para os valores de k e h não estipulam, necessariamente, os melhores valores mas podem, muito bem, servir de ponto de partida para procurar o melhor valor.

Os resultados, que parecem contraditórios são os do método que estima a curva de densidade de probabilidade pela curva normal. Note-se que por esse método conseguiram-se bons resultados. No entanto, isso pode ser um tanto estranho, pois as amostras foram geradas com dois pontos de máximo por classe (pontos da Tabela 6.1), mas a curva normal só estima densidade de probabilidade com um único ponto de máximo, que é o ponto médio da classe. Como as amostras são simétricas em relação ao centro do espaço de

características, pode-se notar, pelas Figuras 6.1 e 6.2, que o ponto médio das amostras de ambas as classes é sempre próximo de (0,5; 0,5) para todos os testes. Como a informação do ponto médio não é capaz de distinguir uma classe da outra, a informação capaz de discriminar as classes está na matriz de covariância de cada classe. A distribuição gaussiana da curva normal multi-dimensional é quase equivalente a calcular a distância Mahalanobis (equação 3.25) entre o padrão a ser classificado e o ponto médio das amostras de uma classe. Mantida uma distância Mahalanobis constante do ponto médio, obtém-se uma elipse. Ou seja, a densidade de probabilidade da classe 0 pode ser vista como uma elipse em 45° centrada próxima a (0,5; 0,5), e a densidade de probabilidade da classe 1 pode ser vista como uma elipse em 135° centrada também próxima de (0,5; 0,5). Então, apesar de a densidade de probabilidade de ambas as classes estar centrada praticamente no mesmo ponto, a regra de Bayes (equação 3.1) acaba fazendo uma boa discriminação entre classes para esse caso. Enfim, para esse caso em particular, mesmo tendo um problema cujas densidades de probabilidades reais possuem dois pontos de máximo e estimando a densidade de probabilidade pela curva normal, que só tem um ponto de máximo, as respostas acabaram sendo bem satisfatórias, o que nem sempre deve ocorrer em problemas reais. Mas, para que essa idéia fique mais clara, foi elaborado o teste 10, onde as amostras primárias representam, de certa forma, a lógica "E" (Tabela 6.11).

Tabela 6.11 - Lógica "E"

x_1	x_1	y
0,33	0,33	0
0,66	0,66	1
0,33	0,66	0
0,66	0,33	0

A Figura 6.4 mostra a distribuição das amostras primárias do teste 10 e as amostras geradas da mesma maneira que nos testes de 1 a 9. Foram criadas 100 amostras de exemplo para cada amostra primária com o desvio-

padrão de 0,1. As amostras de teste foram criadas de maneira idêntica.

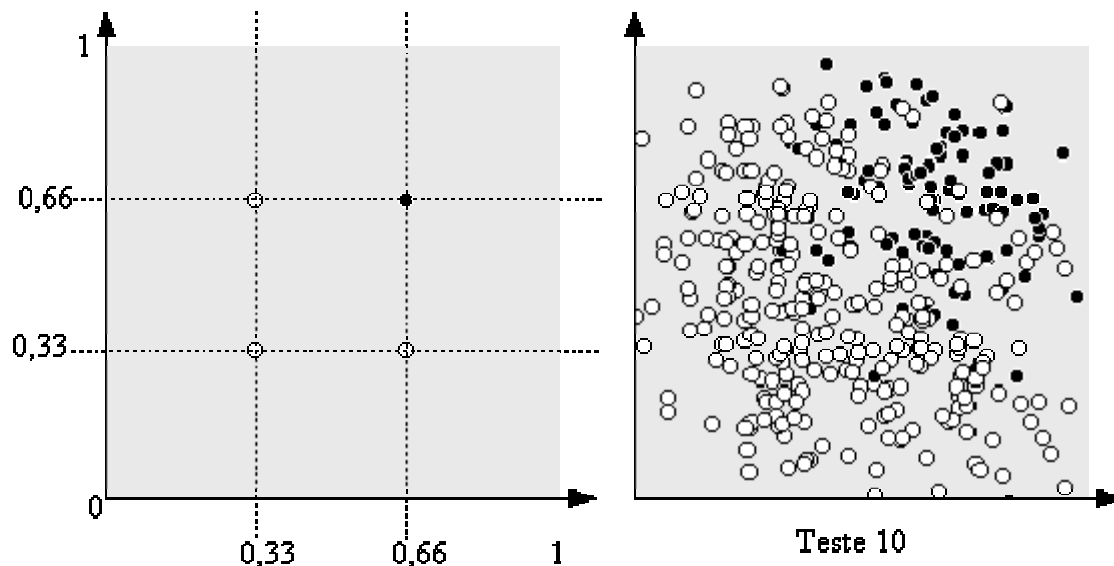


Figura 6.4 - Amostras do teste 10.

Os resultados do teste 10 estão detalhados na Tabela 6.12.

Tabela 6.12 - Teste 10, realizado principalmente para demonstrar a solução de problemas de reconhecimento de padrões pela curva normal, resolve apenas casos específicos.

Teste 10		
Nº de amostras:	404	
Desvio Padrão:	0.1	
	Acerto	Percentual
PID	316	78.22%
SID	327	80.94%
MID	335	82.92%
Elipsoidal	340.5	84.28%
Normal	272.5	67.45%
k-NN	352.5	87.25%
Parzen W.	331	81.93%
RNA	340.5	84.28%

Elipsoidal - Nº de Elipsóides:	49
k-NN - k:	20
Parzen W. - h:	0.22
RNA 2-10-1 Erro <	0.2

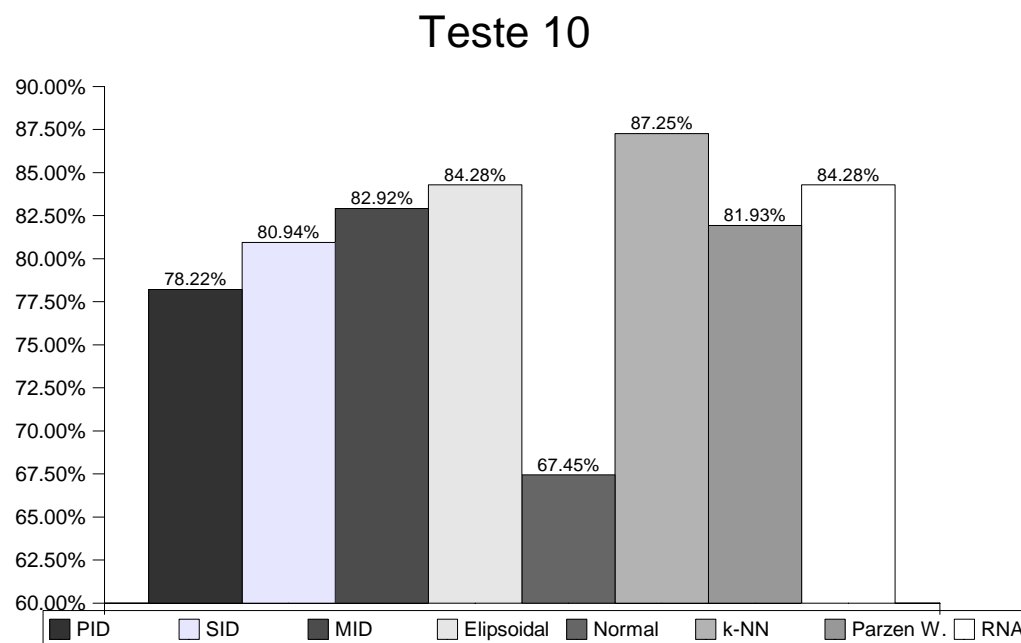


Gráfico 6.10 - Percentual de acerto do teste 10, cujo resultado foi muito diferente dos testes de 1 a 9; isso mostra que para cada tipo de problema de reconhecimento de padrões pode haver um tipo de solução diferente, não havendo uma solução boa para todos os tipos de problemas.

Note-se que, para esse caso, o método de estimar a densidade de probabilidade pela curva normal não teve um bom desempenho. Isso porque, com essa distribuição de amostras, a densidade de probabilidade não pode ser bem estimada pela curva normal. Comparando os mapas de resposta da Figura 6.5, pode notar-se que o mapa da curva normal não teve muita coerência com o conjunto de amostras.

Os métodos PID, SID e MID tiveram um bom desempenho. Nos testes de 1 a 9 o que mais se destacou foi o PID, em seguida, o SID e, por último, o MID. Já no teste 10 o método que teve melhor desempenho foi o MID seguido do SID e, por último, do PID. Entre os métodos PID, SID e MID não se pode concluir qual é o melhor, mas sim que eles têm características distintas. Observando os mapas de resposta das Figuras 6.3 e 6.5, pode notar-se que o método PID, praticamente ignora as amostras de uma classe que estão no meio de muitas amostras da outra classe, enquanto o método MID sempre considera bem todas as amostras. Já o método SID tem características sempre intermediárias entre os métodos PID e MID.

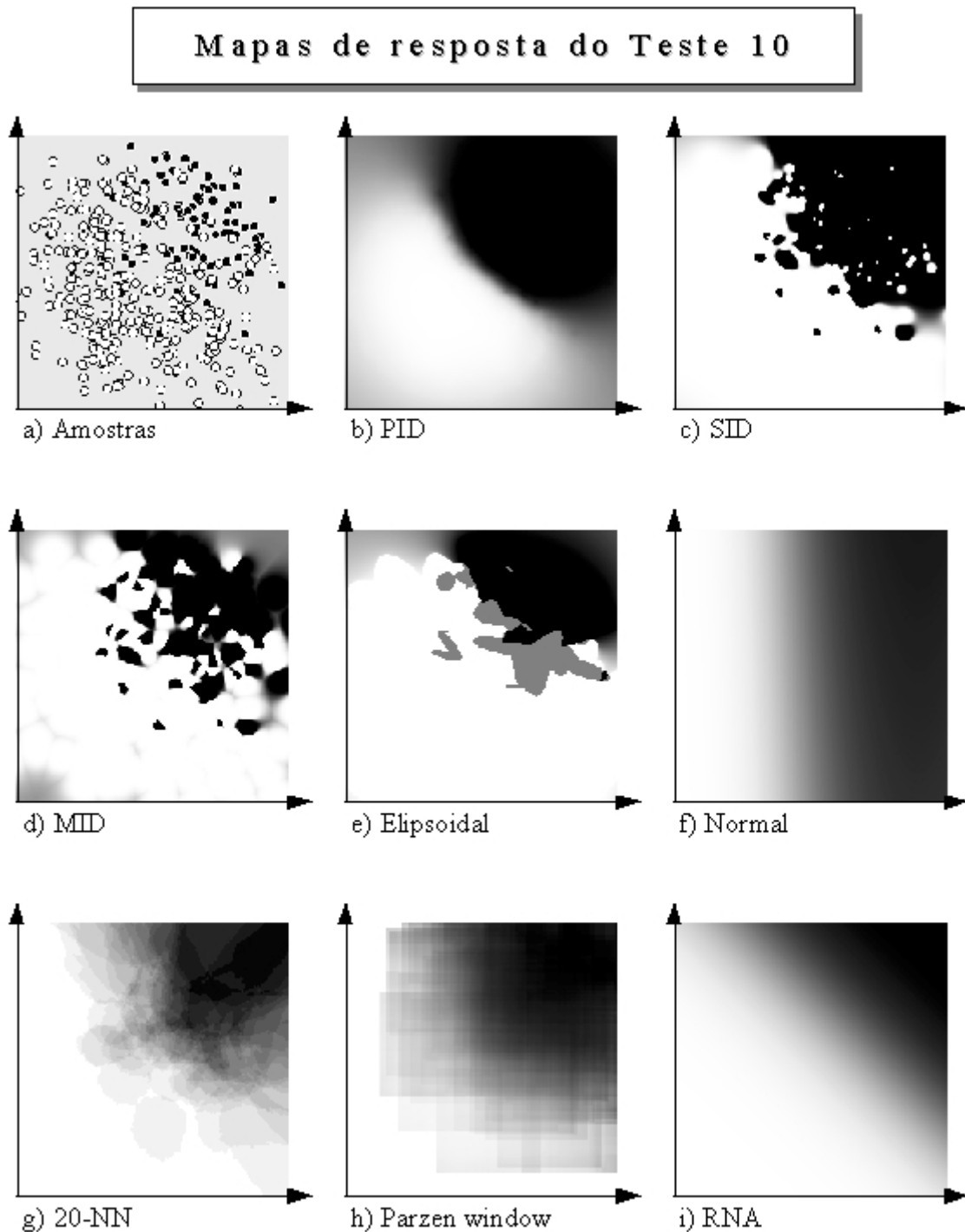


Figura 6.5 - Mapas de resposta do teste 10.

O método elipsoidal apresentou um desempenho tão bom quanto os métodos PID, SID e MID. É de se esperar que o elipsoidal tenha um desempenho próximo ou superior ao método que foi usado para encontrar as

elipses. Os testes realizados de 1 a 9 sempre utilizaram o PID no algoritmo para encontrar as elipses porque foi este que apresentava as melhores respostas nestes testes. A proximidade que é usada para verificar se a elipse é válida também foi mantida constante em 0,85. Para a execução foi utilizado o PID elipsoidal (equação 4.18). A precisão do cálculo foi padronizada em 30, sendo que o ideal seriam valores mais altos para a precisão. No entanto, quanto maior a precisão do cálculo, maior é o custo computacional. Todos esses parâmetros foram fixados para que se pudesse fazer uma comparação entre os testes, utilizando as mesmas condições. Porém, talvez, ajustando-se outros parâmetros, se possa obter respostas melhores para o método elipsoidal. O algoritmo de busca dos elipsóides conseguiu reduzir a complexidade computacional da classificação. Na média foi encontrada uma elipse para cada cinco amostras. A complexidade computacional do PID, MID e SID é $O(N)$; já a complexidade computacional do método elipsoidal é $O(Ne)$, sendo Ne o número de elipsóides encontrados. Como o *overhead* do método elipsoidal é o dobro dos métodos PID, SID e MID, a complexidade computacional do método elipsoidal é, aproximadamente, dois quintos da complexidade computacional dos métodos PID, SID e MID. Isso mostra que o método elipsoidal é capaz de otimizar os métodos PID, SID e MID.

O teste 11 foi elaborado para evidenciar as diferenças entre os métodos propostos. O conjunto de amostras para o teste 11 foi criado manualmente com o software mapeamento. Foram feitas 21 amostras para a classe preta ($y=1$) e 13 amostras para a classe branca ($y=0$), conforme Figura 6.6a.

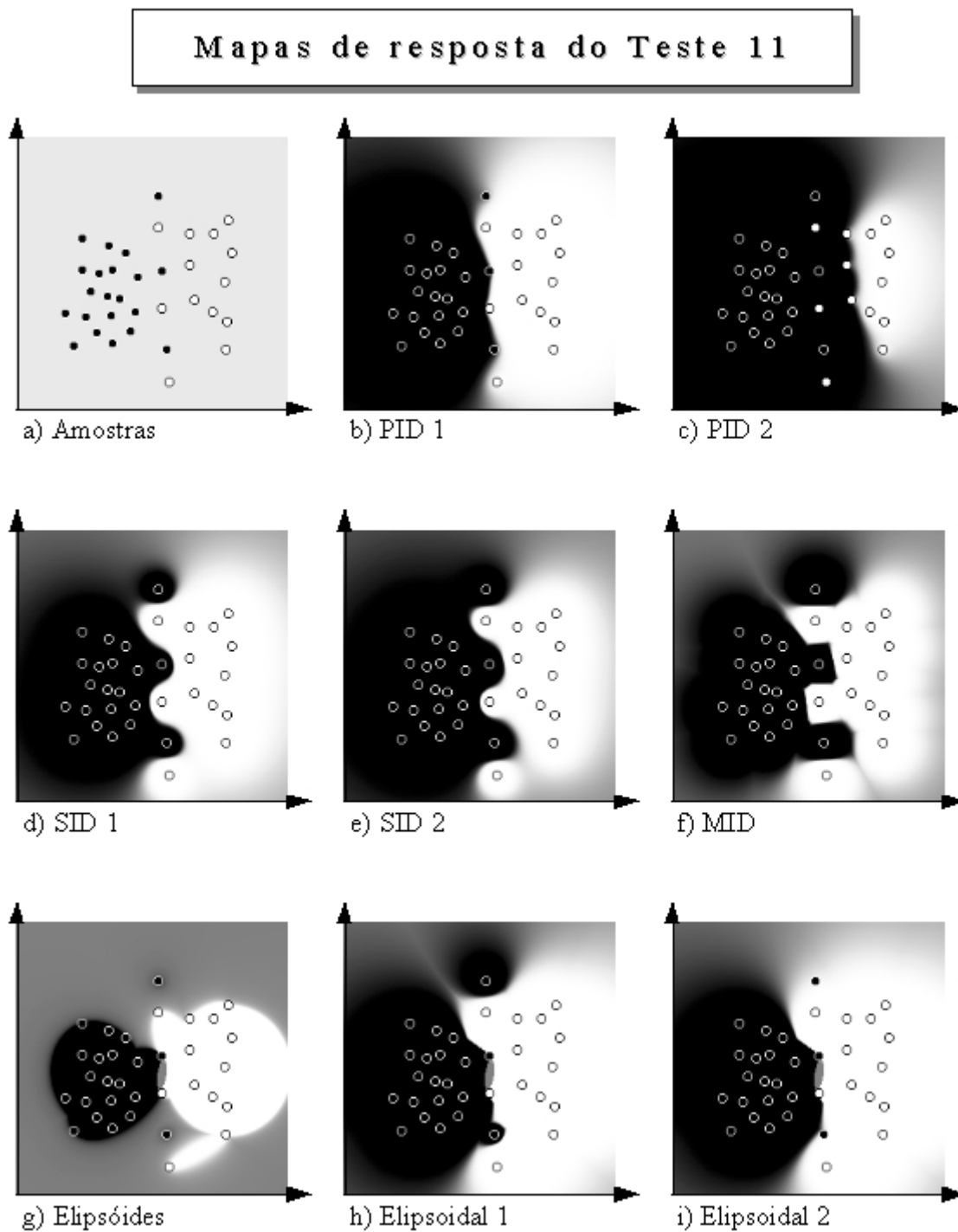


Figura 6.6 - Mapas de resposta do teste 11

A Figura 6.6b apresenta o mapa de resposta do método PID com o grau de generalização igual a 1 ($g=1$ equação 4.5), sem considerar a probabilidade *a priori* (equação 4.9). Já na Figura 6.6c está o mapa de resposta do método PID,

também com o grau de generalização igual a 1, mas considerando a probabilidade *a priori* (equação 4.10). Ou seja, como há mais amostras da classe preta que da classe branca, quando se considera a probabilidade *a priori*, a resposta do PID tende a ser muito mais para a classe preta que para a classe branca.

A Figura 6.6d mostra o mapa de resposta do método SID com o grau de generalização ajustado para 0,3, sem considerar a probabilidade *a priori* (equação 4.11). No mapa da Figura 6.6e está o método SID com o mesmo grau de generalização, mas considerando a probabilidade *a priori* (equação 4.12).

Os quatro mapas de 6.6b a 6.6e procuram evidenciar a diferença na classificação, quando se considera a probabilidade *a priori*. É interessante observar que o método PID é muito mais sensível à probabilidade *a priori* que o método SID. Nos testes de 1 a 9 não importava se a probabilidade *a priori* era considerada, pois havia sempre a mesma quantidade de amostras de cada classe. Assim, a equação 4.9 equivale à equação 4.10 e a equação 4.11 equivale à equação 4.12 ($Q = R$). Já no teste 10, considerar ou não a probabilidade *a priori* faz muita diferença, já que há três vezes mais amostras da classe branca ($y=0$) que da classe preta ($y=1$). Para os teste 10 não se considerou a probabilidade *a priori*.

O mapa de resposta do método MID (Figura 6.6f) foi configurado com o grau de generalização em 0,05. Note-se que a fronteira de decisão do MID é composta por retas, enquanto no PID e no SID é curva. Isso ocorre porque o método MID considera apenas a amostra mais próxima. Deve-se observar que, do gráfico 6.6b ao 6.6f, o grau de generalização foi diminuindo, sendo o efeito disso o aumento da regiões de classificação duvidosa (região cinza).

Os elipsóides foram encontrados, usando-se o método PID para verificar a validade dos elipsóides com o limiar de proximidade ajustado para 0,90 e a precisão do cálculo em 1000. A Figura 6.6g mostra os elipsóides encontrados. As figuras 6.6h e 6.6i exibem o mapa de resposta do método MID elipsoidal (equação 4.21). A diferença é que no primeiro consideram-se as

amostras que não foram englobadas por nenhum elipsóide e, no segundo, as amostras isoladas são ignoradas. Nesse caso, ocorreu uma intersecção entre dois elipsóides de classes opostas. Pelo algoritmo de busca de elipsóides, isso pode ocorrer, mas na região de intersecção nunca existirá amostra de nenhuma das classes, a não ser que a precisão dos cálculos seja ajustada para um valor muito baixo.

7 Estudo de caso

O objetivo deste estudo de caso é fazer a comparação entre os métodos, buscando solucionar um problema real de reconhecimento de padrões. Na mesma época em que foi desenvolvida esta pesquisa, Fábio Iaione pesquisava as alterações provocadas pela hipoglicemia nos sinais de eletroencefalograma (EEG). Iaione coletou sinais EEGs, conhecendo o estado da glicemia do paciente. De intervalos de 2,1 segundos foram extraídas 64 características de frequência dos sinais coletados. De cada intervalo foi obtida uma amostra, em um total de 729 amostras com padrões de hipoglicemia e 729 amostras com padrões normais. Foram separadas 1200 amostras de exemplo e 258 amostras de teste.

As 64 características de frequência foram usadas como vetor de característica; para isso as frequências foram normalizadas ficando entre o intervalo de 0 a 1. Assim, a dimensão de entrada do classificador foi 64. A saída ficou com apenas uma dimensão; se $y = 0$, está em hipoglicemia, se $y = 1$, está normal. No entanto, como a saída do classificador não é binária e varia de 0 a um, foi estabelecido um limiar para cada método, ou seja, quando $y > \text{limiar}$, é classificado como glicemia normal, já se $y \leq \text{limiar}$, classifica-se como hipoglicemia. Para todos os métodos procurou-se o valor de limiar que maximizasse o percentual de acerto.

Os resultados obtidos estão exibidos na Tabela 7.1 e no Gráfico 7.1. O algoritmo de busca dos elipsóides reduziu 1200 amostras para 211 elipsóides. O método k-NN foi executado com $k=35$. O método Parzen window foi executado com $h=0,6$. A configuração da rede neural foi de 64 neurônios na primeira camada mais o bias, 64 neurônios na segunda camada e um neurônio na camada de saída; a rede foi treinada até o erro quadrático médio ficar menor que 0,1.

Tabela 7.1 - Resultado do estudo de caso. Quando $y > \text{limiar}$ classificou-se como glicemia normal; quando $y \leq \text{limiar}$, classificou-se como hipoglicemia.

Estudo de caso		
Nº de amostras de exemplo:	1200	
Nº de amostras de teste:	258	
	Limiar	Percentual
PID	0.58	89.92%
SID	0.6	89.92%
MID	0.36	79.07%
Elipsoidal	0.44	87.21%
k-NN	0.34	87.21%
Parzen W.	0.52	78.68%
RNA	0.3	93.02%

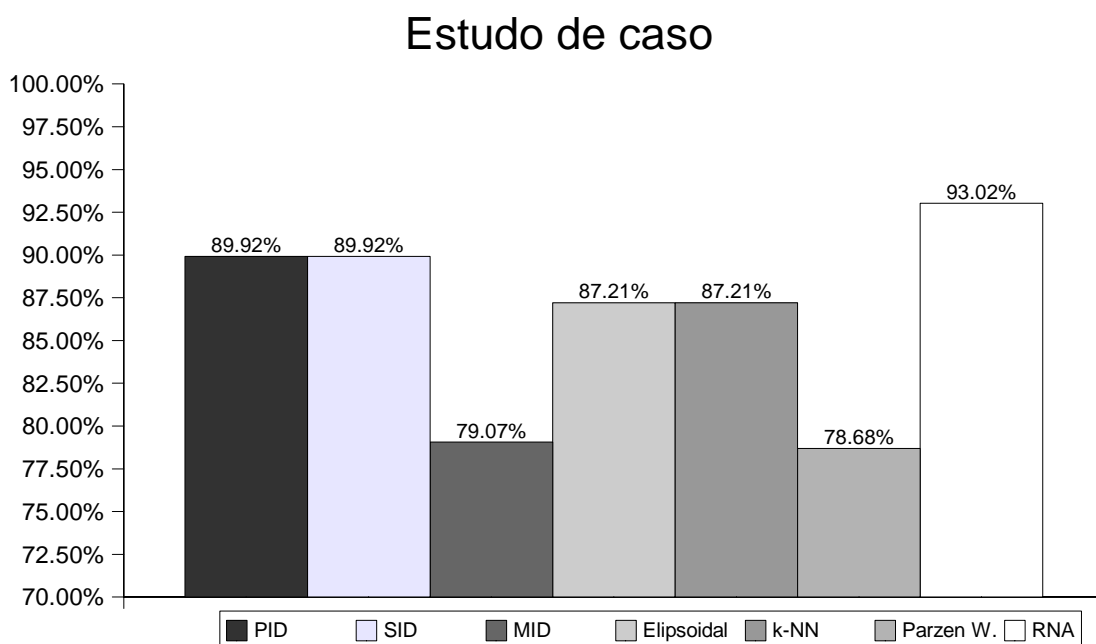


Gráfico 7.1 - Percentual de acerto da detecção da hipoglicemia.

Neste estudo de caso o melhor desempenho foi com a rede neural. Excetuando o Parzen window, que obteve o pior desempenho, e o MID, todos os métodos obtiveram um resultado satisfatório. A rede neural teve um melhor desempenho neste caso por não ter dependência métrica, ou seja, das 64 características extraídas do sinal certamente há entre elas algumas que não são relevantes para a classificação. Essas características mais atrapalham do

que ajudam na classificação por métodos métricos. No entanto, durante o treinamento de uma rede neural, os parâmetros que atrapalham passam a ter pouca influência. O método MID obteve um desempenho baixo provavelmente pelo grande número de amostras de exemplo. Como esse método considera somente a amostra mais próxima, é mais adequado, quando o número de amostras é relativamente baixo.

O método de classificação pela curva normal foi inviável devido à alta dimensão do vetor de características. Os valores dos determinantes das matrizes de covariância eram praticamente nulos, o que impossibilitou a classificação. Os resultados das matrizes de covariância indicam que entre as 64 características há algumas muito correlacionadas.

8 Discussão

Dos métodos estatísticos estudados, verificou-se que os métodos paramétricos não são muito fáceis de aplicar. O problema básico no projeto de sistemas de reconhecimento estatístico de padrões é conseguir estimar a densidade de probabilidade. Conhecendo-se ao menos uma estimativa da densidade de probabilidade, pela regra de Bayes calcula-se a probabilidade *a posteriori*. Ao todo foram apresentadas, aqui, cinco formas de se estimar a densidade de probabilidade, sendo três delas paramétricas.

Pode-se estimar a densidade de probabilidade de modo paramétrico, quando já se conhece a forma da função, mas não os seus parâmetros. Nesse caso, utiliza-se a máxima verossimilhança ou aproximação bayesiana. Na maioria das vezes, esses dois métodos se equivalem. Mas ambos exigem que se conheça uma função que sirva para estimar a densidade de probabilidade. Na prática, é muito complicado encontrar tal função, já que a sua integral, no espaço de característica, deve ser 1, e a maioria das funções conhecidas que obedecem a essa condição são unimodais, tal como a curva normal. Enfim, encontrar uma função para estimar a densidade de probabilidade e estimar os parâmetros usando máxima verossimilhança ou aproximação Bayesiana é uma tarefa muito complicada e com poucas chances de bom desempenho.

A primeira forma paramétrica assume que a distribuição de amostras é normal. Esse método foi aplicado no estudo comparativo e mostrou-se eficiente apenas para casos específicos. Na maioria dos casos reais de reconhecimento de padrões, estimar densidade de probabilidade de uma classe por uma única curva normal é insuficiente. Além disso, a curva normal precisa de uma matriz de covariância cuja dimensão é D^2 e, para problemas cuja dimensão é elevada, trabalhar com essa matriz pode ser computacionalmente custoso, tanto em memória quanto em processamento. Uma das grandes vantagens dos métodos paramétricos é que se consegue reduzir o custo computacional de processamento e memória durante a classificação, o que deixa, todavia, de ser válido para a curva normal em problemas de alta dimensão.

Já os métodos estatísticos não paramétricos são bastante eficientes. O k-NN é um método intuitivo, simples, robusto e popular, embora nem sempre associado a métodos estatísticos. Sua principal vantagem é estimar diretamente a probabilidade *a posteriori*, não sendo preciso estimar a densidade de probabilidade, apesar de possível, se necessário. Em virtude de sua popularidade, foram desenvolvidos vários métodos para otimizá-lo [BANDYOPADHYAY 2001][WU 2001].

O método Parzen window estima a densidade de probabilidade de forma muito similar ao k-NN. Também é eficiente e relativamente simples. Uma diferença que pode ser notada, na prática, entre o k-NN e o Parzen window: para padrões relativamente distantes de qualquer amostra, o Parzen window faz a classificação, dando pouca certeza, enquanto o k-NN tende a dar certeza mesmo a padrões longe de qualquer amostra. Essa característica pode ser observada no estudo comparativo desenvolvido no capítulo 6.

Em alguns testes, a rede neural perceptron multicamadas apresentou um desempenho fraco por dificuldade de treinamento. Esse problema ocorre, quando o grau de dispersão das amostras o conjunto de treinamento são muito grandes. Uma solução simples para esse problema seria submeter as amostras de treinamento ao algoritmo de aglomeração k-médias. Assim, se reduz o conjunto de amostras e se espera que também se reduza o grau de dispersão, tornando mais fácil o treinamento da rede. Por outro lado, no estudo de caso, ficou evidente o mérito da rede neural de não ter dependência métrica. Enquanto para os métodos com dependência métrica o fator de escala de cada característica de entrada é decisivo para uma boa classificação, nas redes neurais esse problema não existe.

Uma rede neural pode ser treinada com amostras cujo vetor de saída tem valores contínuos, enquanto todos os outros métodos podem utilizar somente amostras com o vetor de saída binário. Nesse caso, as aplicações para rede neural vão além da classificação de padrões.

O estudo comparativo e de caso mostrou que para cada aplicação pode

haver um método de reconhecimento de padrões mais adequado. Apesar de os métodos propostos obterem um desempenho inferior que ao da rede neural no estudo de caso, podem ser considerados como métodos válidos, pois no estudo de comparativo há testes em que os métodos propostos tiveram um desempenho melhor. Sendo assim, os métodos propostos são uma alternativa a considerar para o projeto de sistemas de reconhecimento de padrões. Cada um possui um comportamento que pode adequar-se melhor a cada situação. O método PID parece melhor, quando se tem uma grande quantidade de amostras com alto grau de dispersão. Já o MID funciona bem, quando o grau de dispersão das amostras é pequeno e o conjunto de amostras não é tão grande. O método SID tem sempre características intermediárias entre o PID e o MID.

Todos os métodos propostos possuem uma maneira fácil de arbitrar o grau de generalização. Para alguns sistemas de reconhecimento de padrões isso pode ser muito interessante. Por exemplo, um sistema implementado para sugerir um diagnóstico médico deve ser configurado com um grau de generalização relativamente baixo, pois um padrão longe das amostras é classificado como duvidoso, ou seja, se o sistema não conhece padrão parecido, não sugere diagnóstico, evitando sugestões errôneas. Mesmo não havendo uma metodologia bem definida para encontrar um grau de generalização adequado, essa tarefa é simples de realizar na prática.

A complexidade computacional dos métodos PID, SID e MID é proporcional ao número de amostras ($O(N)$). No entanto, o método MID, que só utiliza a menor distância, equivale ao 1-NN. Assim, os mesmos algoritmos de otimização desenvolvidos para o k-NN podem ser aplicados ao MID.

O método elipsoidal mostrou ter um bom desempenho de classificação e otimizou os três métodos referidos. No entanto, ainda é necessário desenvolver melhor o método de busca dos elipsóides. Da maneira como foi implementado, esse método tem um custo computacional muito elevado - cerca de $O((N^3 \cdot P)/8)$, sendo P a precisão do cálculo. Esse custo torna tedioso o

trabalho de procurar a melhor configuração dos parâmetros para o algoritmo de busca. Além disso, nada garante que os elipsóides encontrados sejam os melhores. Na realidade, esse algoritmo foi desenvolvido, pensando-se em uma distribuição de amostras com baixo grau de dispersão. Acredita-se que, melhorando o algoritmo de busca, correlatamente o método elipsoidal ficaria ainda mais otimizado e com um desempenho superior.

9 Conclusão

O primeiro objetivo do trabalho, fornecer uma breve introdução ao estudo de técnicas de reconhecimento de padrões, foi cumprido. Foram apresentadas as principais técnicas de reconhecimento de padrões: regra de Bayes, máxima verossimilhança, aproximação bayesiana, vizinhança mais próxima (k-NN), Parzen window, perceptron multicamadas, redes RBF e mapas de Kohonen.

Ao todo foram desenvolvidos quatro métodos de reconhecimento de padrões, três deles não paramétricos (PID, SID, MID) e o quarto método, elipsoidal, paramétrico, que otimiza os três primeiros.

A comparação feita entre alguns métodos tradicionais e os aqui desenvolvidos mostrou que estes tiveram um bom desempenho e se apresentaram como mais uma possibilidade a ser considerada em projetos de sistemas de reconhecimento de padrões. A Tabela 9.1 contém um resumo da comparação feita entre os métodos.

Tabela 9.1 - Resumo da comparação entre os métodos de reconhecimento de padrões.

Método	Comentários
Curva normal	Assume que a densidade de probabilidade é unimodal. Funciona bem somente para casos específicos. Não é adequado para problemas com altas dimensões de entrada. Tem dependência métrica.
Parzen Window	Estima a densidade de probabilidade diretamente do conjunto de amostras. Não necessita de treinamento. A complexidade computacional aumenta diretamente com o número de amostras. Tem dependência métrica.
k-NN (Vizinhança mais próxima - Nearest Neighbor)	Estima a probabilidade <i>a posteriori</i> diretamente do conjunto de amostras. Não necessita de treinamento. A complexidade computacional aumenta diretamente com o número de amostras. Tem dependência métrica.
RNA (Redes Neurais Artificiais perceptron multi-camadas)	Necessita de treinamento. Tem baixa complexidade computacional. Não tem dependência métrica.
PID (Produtório do Inverso das distâncias)	Não necessita de treinamento. A complexidade computacional aumenta diretamente com o número de amostras. Geralmente tem bom desempenho com grande quantidade de amostras com alto grau de dispersão. Tem dependência métrica.
SID (Somatório do Inverso das distâncias)	Não necessita de treinamento. A complexidade computacional aumenta diretamente com o número de amostras. Tem características intermediárias entre os métodos PID e MID. Tem dependência métrica.
MID (Mínima distância)	Não necessita de treinamento. A complexidade computacional aumenta diretamente com o número de amostras. Geralmente tem bom desempenho com poucas amostras de baixo grau de dispersão. Tem dependência métrica.

<i>Método</i>	<i>Comentários</i>
Elipsoidal	Otimiza os métodos PID, SID e MID. Necessita de treinamento. A complexidade computacional depende do resultado do treinamento. Tem dependência métrica.

O software *classificador* mostrou ser uma ferramenta útil, pois possibilita testar, em um mesmo programa, diferentes técnicas, e, assim, verificar e escolher qual a mais adequada para o problema. O software *mapeamento* também se mostrou bastante útil, mapeando as respostas de cada método; no entanto, só tem sentido utilizá-lo, quando a dimensão de entrada tem apenas duas características.

Finalmente, fica como sugestão pretensiosa batizar as técnicas aqui propostas de método Burckas de reconhecimento de padrões.

10 Futuros trabalhos

Como futuro trabalho propõe-se desenvolver um algoritmo mais rápido para buscar os elipsóides que possibilitem um melhor desempenho de classificação para o método elipsoidal.

Pode-se desenvolver um algoritmo de classificação não supervisionado, usando a mesma idéia do método elipsoidal.

Ainda podem ser implementados mais métodos de reconhecimento ao software *classificador*, como k-médias, redes RBF e mapas de Kohonen.

11 Bibliografia

- [ARGOUD 2001]: ARGOUD, Fernanda Isabel Marques. *Contribuição à Automatização da Detecção e Análise de Eventos Epileptiforme*. UFSC, 2001. Tese - (Doutorado em Engenharia Biomédica) - Programa de pós-graduação em Engenharia Biomédica - Universidade Federal de Santa Catarina.
- [BANDYOPADHYAY 2001]: BANDYOPADHYAY, S.; MAULIK, U. *Efficient prototype reordering in nearest neighbor*. Pattern Recognition, India, vol. 35, 2002.
- [CANTÚ 1997]: CANTÚ, Marco. *Delphi 3 "a bíblia"*. Primeira edição. Brasil: Makron Books do Brasil, 1997.
- [de AZEVEDO 2000]: de AZEVEDO, Fernando M.; BRASIL, Lourdes M.; de OLIVEIRA, Roberto C. L. *Redes Neurais com Aplicações em Controle e em Sistemas Especialistas*, Primeira edição. Brasil: Visual Books, 2000.
- [DUDA 2001]: DUDA, R.O.; HAR, P.E.; STORK, D.G. *Pattern Classification*. Segunda edição. USA: Wiley-Interscience, 2001.
- [GARCIA 2001]: GARCIA, Euler de Vilhena. Instrumentação para a monitoração das alterações eletrocardiográficas decorrentes da hipoglicemia. Florianópolis, 2001. 72 f. Dissertação (Mestrado) - Universidade Federal de Santa Catarina.
- [HART 1967]: COVER, T. M.; HART, P. E. *Improve k-nearest neighbor classification*, IEEE Transactions on Information Theory, vol. it-13, no 1, jan/1967.
- [IAIONE 2002]: IAIONE, Fábio; MARQUES, Jefferson L. B. *Desenvolvimento de um Sistema para Registro e Análise do EEG Aplicado à Detecção de*

- Hipoglicemia*. In: Congresso Brasileiro de Engenharia Biomédica (18.: Set 2002: São José dos Campos-SP). Anais v. 3/5, p. 70 -73.
- [JAIN 1988]: JAIN, A.K.; RAMASWAMI,M.D. *Classifier Design with Parzen Windows*, North-Holland, 1988.
- [JAIN 2000]: JAIN, K.Anil; DUUIN, Robert P.W.; MAO, Jianchang. *Statistical Pattern Recognition: A Review*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 1 jan/2000.
- [MOREIRA 2002]: MOREIRA, Fabiano Cordeiro. *Reconhecimento e Classificação de Padrões de Imagens de Núcleos de Linfócitos do Sangue Periférico Humano com a Utilização de Redes Neurais*. Brasil, 2002. Dissertação (Mestrado em Ciência da Computação) - Programa de pós-graduação em ciência da computação, Universidade Federal de Santa Catarina.
- [O'BRIEN 1996]: O'BRIEN, Stephen. *Turbo Pascal 6 completo e total*. Primeira edição. Brasil: Makron Books do Brasil, 1996.
- [PATRICK 1972]: PATRICK, Eduard A. *Fundamentals of Pattern Recognition*. Primeira edição. USA: Prentice-Hall, 1972.
- [SILVA 1998]: SILVA, Junior, Silvio Moraes; AZEVEDO, Fernando Mendes de; MARINO NETO, Jose. Sistema microcontrolado de estimulação e análise de potenciais evocados para utilização com eletroencefalografia computadorizada. 1998 129f. Dissertação (Mestrado) - Universidade Federal de Santa Catarina.
- [SOUZA 1999]: DE SOUZA,J.A. *Recohecimento de Padrões Usando Indexação Recursiva*. UFSC, 1999. Tese - (Doutorado em Engenharia de Produção e sistemas) - Programa de pós-graduação em Engenharia de Produção -

Universidade Federal de Santa Catarina.

[SWOKOWISKI 1990]: SWOKOWISKI. *Cálculo com geometria analítica*. Segunda edição. Brasil: Makron Books do Brasil, 1990.

[TODESCO 1995]: TODESCO, José Leomar. *Reconhecimento de padrões usando rede neuronal artificial com uma função radial de base: Uma aplicação na classificação de cromossomos humanos*, Brasil, 1995. Tese - (doutorado em Engenharia de Produção e Sistemas) - Programa de pós-graduação em Engenharia de Produção - Universidade Federal de Santa Catarina.

[UHR 1973]: UHR, Leonard. *Pattern Recognition, Learning, and Thought*. Primeira edição. USA: Prentice-Hall, 1973.

[WU 2001]: WU, Y.; IANAKIEV, K.; GOVINDARAJU, V. *Improve k-nearest neighbor classification*. Pattern Recognition, USA, vol. 35, 2002.