



**COPPE/UFRJ**

**IDENTIFICAÇÃO DE FALHAS EM BOMBEIO MECÂNICO A PARTIR DE  
CARTAS DINAMOMÉTRICAS DE FUNDO**

**Nathalia Marassi Cianni**

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientador: Geraldo Bonorino Xexéo

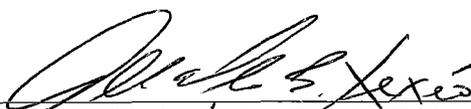
Rio de Janeiro  
Março de 2009

IDENTIFICAÇÃO DE FALHAS EM BOMBEIO MECÂNICO A PARTIR DE  
CARTAS DINAMOMÉTRICAS DE FUNDO

Nathalia Marassi Cianni

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

  
Prof. Geraldo Bonorino Xexéo, D.Sc.

  
Prof. Luís Alfredo Vidal de Carvalho, D.Sc.

  
Prof.ª Adriana Santarosa Vivacqua, D.Sc.

RIO DE JANEIRO, RJ - BRASIL

MARÇO DE 2009

Cianni, Nathalia Marassi

Identificação de Falhas em Bombeio Mecânico a partir de Cartas Dinamométricas de Fundo/ Nathalia Marassi Cianni. – Rio de Janeiro: UFRJ/COPPE, 2009.

XI, 129 p.: il.; 29,7 cm.

Orientador: Geraldo Bonorino Xexéo

Dissertação (mestrado) – UFRJ/ COPPE/ Programa de Engenharia de Sistemas e Computação, 2009.

Referências Bibliográficas: p. 97-101.

1. Redes Neurais Artificiais. 2. Cartas Dinamométricas de Fundo. 3. Bombeio Mecânico. I. Xexéo, Geraldo Bonorino. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

*Dedico esta dissertação  
aos meus pais Celso (in memoriam) e Sônia,  
aos meus irmãos Christiano e Edmundo  
e às minhas sobrinhas Luíza e Beatriz,  
por existirem em minha vida.*

# Agradecimentos

Agradeço ao Luiz Alberto e ao Xexéo por terem apoiado, desde o início, a minha vontade em aplicar a pesquisa de excelência da COPPE ao aperfeiçoamento do trabalho de uma das maiores empresas de energia do mundo, a Petrobras.

Ao Rutácio e ao Benno, por terem sugerido o tema desta dissertação, pelas aulas de Bombeio Mecânico e, principalmente, pela atenção com que sempre atendem a mim e a toda equipe SOLAR.

Ao Gustavo, por ter confiado este tema a mim, pelas explicações e ajuda na classificação das cartas e por todo o incentivo.

Aos professores Luis Alfredo de Carvalho e Adriana Vivacqua, por aceitarem participar da banca de avaliação desta dissertação, mesmo diante de tantos compromissos e de um convite um tanto em cima da hora.

Ao Fernando Rodrigues, por ir além do seu papel de gerente, fornecendo idéias e até implementando modelos matemáticos para contribuir para a solução do problema apresentado neste trabalho.

Aos queridos amigos Luiz Alberto, Bruno e Edmundo que, além do incentivo, participaram da tarefa mais cansativa: a revisão do texto. Muito obrigada pela atenção com que revisaram cada palavra e por todas as sugestões.

Aos amigos do SOLAR e da SNEP que tanto me ajudaram, assumindo os projetos que tive que abandonar temporariamente para terminar esta dissertação, oferecendo livros e material de apoio, emprestando a máquina para rodar os testes, e, não menos importante, participando dos momentos de descontração, tão importantes para elevar o astral de um mestrando.

Ao Blaschek, pelo exemplo e apoio fundamental no início do mestrado.

Aos professores da graduação (UFRJ) e do mestrado (PESC/COPPE), responsáveis pela minha formação e que contribuem para o ensino de excelência desta universidade.

Aos amigos da pós, em especial à Carol, pelas várias discussões sobre o tema desta dissertação e sobre os *toolboxes* do Matlab.

Aos amigos da ETPC, do Fundão, da COPPETEC, da Universidade Petrobras e do mestrado (em especial, à Diretoria! Ainda de pé?), pelos momentos de alegria que tornam nossa vida mais fácil. Em especial também à Fefs, pela lealdade à nossa amizade e por estar sempre presente, nos melhores e piores momentos, desde a Escola Técnica; e, ao Rafa, por toda a ajuda e companheirismo.

Ao meu saudoso pai, por todo amor e orgulho que sentia de nós, seus filhos, fazendo-nos acreditar que poderíamos ir adiante. À minha mãe, pelo incentivo, pela confiança e pelo exemplo de amor ao próximo ao dedicar sua vida não só à sua família, mas a todos que, dela, precisam. Aos meus irmãos, pelo exemplo de perseverança, pelo carinho e cuidado que sempre tiveram comigo e por me darem as sobrinhas mais encantadoras do mundo! E, finalmente, a elas, minhas sobrinhas, e aos demais pequenos da nova geração da família (e não são poucos!), que, com toda a ingenuidade e inteligência, vieram nos mostrar o quão pouco é preciso para sermos felizes.

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

## IDENTIFICAÇÃO DE FALHAS EM BOMBEIO MECÂNICO A PARTIR DE CARTAS DINAMOMÉTRICAS DE FUNDO

Nathalia Marassi Cianni

Março/2009

Orientador: Geraldo Bonorino Xexéo

Programa: Engenharia de Sistemas e Computação

O bombeio mecânico é o método de elevação artificial de óleo mais utilizado no mundo. A rápida identificação de falhas em sua operação é essencial para reduzir as despesas com manutenção corretiva nas unidades de bombeio e aumentar a produção de óleo. A operação de bombeio em um poço pode ser acompanhada através da observação do formato da carta dinamométrica de fundo (CDF). A CDF é um gráfico, gerado a cada ciclo de bombeio, que apresenta a variação das cargas que atuam sobre a bomba de fundo durante o seu deslocamento. A interpretação da carta, no entanto, é uma tarefa desgastante e demorada. Este trabalho propõe uma rede neural supervisionada para realizar o reconhecimento automático da falha a partir do formato da carta. A aplicabilidade de outros tipos de redes neurais ao problema exposto também é apresentada. Como produto final deste estudo, foi implementado um módulo capaz de diagnosticar falhas em sistemas de bombeio mecânico a partir da classificação da CDF realizada pela rede neural proposta.

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

IDENTIFICATION OF ROD PUMPING FAULTS FROM DOWNHOLE  
DYNAMOMETER CARDS

Nathalia Marassi Cianni

March/2009

Advisor: Geraldo Bonorino Xexéo

Department: Systems Engineering and Computer Science

Sucker rod pumping is the most widely used method of artificial lift in the world. Identifying rod pumping faults as soon as possible is essential to increase oil production and to reduce operating costs for corrective maintenance. Pumping conditions may be monitored through the shape of a load vs. position plot of the pump, known as downhole dynamometer card (DDC). The interpretation of the DDC, however, is a hard and time consuming task. This work proposes a supervised neural network to perform automatic recognition of the DDC shape. It also evaluates the applicability of others kind of artificial neural networks to address this problem. As a final product, a software was developed to diagnose faults in rod pumping systems from the DDC classification performed by the proposed neural network.

# Índice

1	Introdução .....	1
2	Métodos de Elevação .....	4
2.1	Sistema de bombeio mecânico (SBM).....	5
2.1.1	Componentes Principais.....	6
2.1.1.1	Equipamentos de Superfície: A Unidade de Bombeio (UB).....	6
2.1.1.2	A Coluna de Produção.....	7
2.1.1.3	A Coluna de Hastes .....	7
2.1.1.4	A Bomba de Fundo .....	9
2.1.2	Cartas Dinamométricas .....	11
2.1.2.1	Carta Dinamométrica de Superfície (CDS).....	12
2.1.2.2	Carta Dinamométrica de Fundo (CDF).....	15
3	Redes Neurais Artificiais .....	21
3.1	Estrutura Básica.....	22
3.1.1	Funções de Ativação .....	24
3.2	Arquitetura de uma Rede Neural.....	25
3.3	Aprendizado de uma Rede Neural .....	26
3.3.1	Forma de apresentação dos dados: Batch x On-line .....	27
3.3.2	Tipos de treinamento.....	28
3.3.2.1	Treinamento Supervisionado.....	28
3.3.2.2	Treinamento Não-Supervisionado.....	29
3.3.3	Redes Perceptron de uma camada.....	30
3.3.4	Redes Multilayer Perceptron (MLP).....	31
3.3.4.1	<i>Backpropagation</i> (Retropropagação) .....	32

3.3.5	Capacidade das redes MLP .....	33
3.3.5.1	Convergência x Generalização.....	34
3.3.5.2	Paralisia da rede .....	35
3.3.6	Comparação entre modelos .....	36
3.3.6.1	K-Fold Cross-Validation.....	37
3.3.6.2	Métricas para comparação de modelos .....	38
3.3.7	Redes Não-Supervisionadas - Competitive Learning .....	40
3.3.7.1	<i>Self-Organizing Maps (SOM)</i> ou Mapas de Kohonen .....	40
3.3.8	Redes ART .....	44
3.3.8.1	Redes Fuzzy-ART .....	45
3.3.8.2	Redes Fuzzy ARTMAP.....	46
4	Trabalhos Relacionados.....	49
5	Metodologia para Definição da Rede Neural .....	59
5.1	Pré-processamento dos dados.....	59
5.1.1	Normalização dos dados.....	59
5.1.2	Redução de atributos .....	60
5.2	Definição da Rede Neural .....	65
5.2.1	Número de camadas escondidas.....	65
5.2.2	Número de neurônios na camada escondida .....	66
5.2.3	Número de saídas da rede.....	66
5.2.4	Função de ativação .....	66
5.3	Treinamento da Rede Neural.....	67
5.3.1	Número de ciclos de treinamento.....	67
5.3.2	Conjunto de treinamento .....	67
5.3.3	Algoritmo de treinamento .....	69
5.4	Avaliação dos Resultados.....	69
5.4.1	Discretização da saída .....	69

5.4.2	Aperfeiçoamento dos resultados .....	70
<b>6</b>	<b>Validação da Rede Neural .....</b>	<b>71</b>
6.1	Definição da amostra de dados utilizada.....	71
6.2	Resultado Inicial.....	72
6.3	Redução de Atributos .....	73
6.4	Discretização das Saídas .....	76
6.5	MLP x Outros Tipos de RNA .....	77
6.5.1	Uma rede com N saídas x N redes com uma saída .....	77
6.5.2	Redes Fuzzy ARTMAP.....	78
6.5.3	Mapas de Kohonen.....	79
6.6	Comparação com os Trabalhos Relacionados.....	82
<b>7</b>	<b>IFBM – Módulo de Identificação de Falhas em Sistemas de Bombeio Mecânico a partir de Cartas Dinamométricas de Fundo ...</b>	<b>85</b>
7.1	SOLAR - Sistema de Otimização e Análise Remota de Dados de Produção .85	
7.2	Módulo de Redes Neurais Artificiais (RNA).....	86
7.2.1	Exemplo do AND de Três Entradas.....	87
7.3	A Arquitetura do IFBM.....	90
7.4	A Interface do IFBM.....	92
<b>8</b>	<b>Conclusões .....</b>	<b>94</b>
8.1	Trabalhos Futuros.....	95
	<b>Referências Bibliográficas .....</b>	<b>97</b>
	<b>Apêndice A.....</b>	<b>102</b>
	<b>Apêndice B .....</b>	<b>122</b>
	<b>Apêndice C .....</b>	<b>127</b>

# 1 Introdução

O bombeio mecânico é o método de elevação artificial de petróleo mais utilizado no mundo (Brown, 1982, Clegg e Buccaram, 1993, Ceylan, 2004). Aplicado somente para extração de óleo terrestre (*onshore*), o sistema de bombeio mecânico está presente em cerca de 60% dos poços brasileiros (Barreto, 1993). A rápida identificação de falhas em sua operação evita a perda de produção de óleo e permite a redução de despesas, já que a manutenção de poços é uma atividade custosa e que despende muito tempo (Derek *et al.*, 1988).

O comportamento da carga que atua sobre a bomba de fundo representa o funcionamento do sistema de bombeio mecânico. Por isso, a carta dinamométrica de fundo – um gráfico que mostra a variação das cargas em função do deslocamento da bomba – é um dos principais recursos utilizados no controle da operação de poços de bombeio mecânico.

Diferentes formatos da carta representam diferentes situações operacionais de bombeio, podendo indicar tanto uma condição normal de bombeio quanto uma falha operacional no sistema de elevação (Xu *et al.*, 2007). No entanto, cartas semelhantes podem ser geradas a partir de situações distintas. Para distinguir situações ambíguas, o especialista precisa recorrer a informações adicionais sobre o fluido extraído, o reservatório de óleo, o esquema mecânico do poço, o histórico de intervenções e o histórico de testes de produção realizados naquele poço. Diagnosticar falhas em um sistema de bombeio mecânico através de cartas não é, portanto, uma tarefa trivial.

A carta dinamométrica é gerada a cada ciclo de bombeio. Um poço que realiza, em média, 10 ciclos por minuto, gera cerca de 14.400 cartas por dia. Tratando-se de um número elevado de poços, a tarefa de acompanhá-los através da visualização da forma de suas cartas torna-se intratável.

Técnicas de Inteligência Artificial têm sido aplicadas para fazer o reconhecimento automático da forma da carta. Na literatura, encontra-se o reconhecimento de padrão de carta por: métodos estatísticos (Foley e Svinos, 1987); métodos numéricos (Keating *et al.*, 1991); método de *grid* (Dickinson e Jennings, 1990); método baseado em séries de Fourier (Dickinson e Jennings, 1990); redes

neurais supervisionadas (Rogers *et al.*, 1990; Nazi *et al.*, 1994); e por redes neurais auto-organizáveis (Xu *et al.*, 2007).

Apesar das pesquisas e trabalhos realizados, o problema do reconhecimento automático das cartas ainda não está bem resolvido e, por isso, permanece como objeto de estudo de universidades e companhias petrolíferas.

O presente trabalho aplica uma rede neural supervisionada, do tipo *Multi-Layer Perceptron* (MLP), para a solução desse problema e propõe algumas abordagens para melhorar a sua eficiência na classificação das cartas através do seu formato. Uma das abordagens propostas consiste em se considerar apenas as coordenadas de carga de uma carta dinamométrica, descartando-se as posições registradas durante o ciclo de bombeio.

Também faz parte do escopo desse trabalho, a comparação entre o desempenho de uma rede MLP com  $N$  saídas (uma para cada tipo de carta) e um conjunto de  $N$  redes com apenas uma saída, onde cada rede é treinada para ser especialista em uma única condição de bombeio. Além disso, apresenta-se um estudo sobre a possível substituição da rede MLP por Redes Fuzzy ARTMAP, redes neurais que são baseadas na Teoria da Ressonância Adaptativa. Avalia-se, ainda, a aplicabilidade dos Mapas Auto-Organizáveis de Kohonen – redes neurais não-supervisionadas – para a classificação das cartas dinamométricas.

O produto final deste trabalho é o desenvolvimento de um módulo de identificação de falhas em bombeio mecânico a partir da classificação de cartas dinamométricas de fundo realizada por uma rede neural.

Com a identificação automática das falhas que ocorrem em um sistema de bombeio mecânico, espera-se obter os seguintes benefícios (diretos e indiretos):

- Reduzir o tempo para se diagnosticar um problema no sistema de bombeio mecânico;
- Reduzir o número de diagnósticos interpretados incorretamente;
- Permitir que todos os poços sejam acompanhados através de suas cartas dinamométricas;
- Aumentar a atenção dada na solução do problema (já que o enfoque no diagnóstico será reduzido);

- Reduzir os custos e a perda de óleo gerados pela manutenção corretiva nas unidades de bombeio;
- Diminuir o número de intervenções nos poços;
- Reduzir a perda de óleo devido à submergência;
- Melhorar importantes índices para companhias petrolíferas: aumentar IEP (Índice de Eficiência da Produção); reduzir IPNI (Índice de Perdas Não-Identificadas) e aumentar TMEF (Tempo Médio Entre Falhas).

O próximo capítulo apresenta o funcionamento básico de um sistema de bombeio mecânico, a geração da carta dinamométrica de fundo e como uma falha no bombeamento do óleo pode ser identificada a partir do formato de uma carta. O capítulo 3 introduz o conceito de redes neurais artificiais, a arquitetura básica de uma rede neural e o seu processo de aprendizagem. Apresentam-se, ainda nesse capítulo, as redes baseadas na Teoria da Ressonância Adaptativa e os Mapas de Kohonen. No quarto capítulo, são apresentadas as abordagens para o reconhecimento da carta presentes na literatura. O capítulo 5 descreve a metodologia utilizada para definição da rede neural adotada nesse trabalho. O resultado do emprego dessa metodologia e a comparação entre diferentes tipos de redes neurais para o problema exposto estão no sexto capítulo. O capítulo 7 descreve o módulo de Identificação de Falhas em Sistemas de Bombeio Mecânico a partir de Cartas Dinamométricas de Fundo. Finalmente, as conclusões deste trabalho são apresentadas no capítulo 8.

## 2 Métodos de Elevação

O petróleo se acumula em rochas porosas e permeáveis, formando os reservatórios. A extração do óleo e de outros fluidos contidos nessas rochas pode ocorrer através da elevação natural dos fluidos ou por métodos artificiais de elevação.

A elevação natural ocorre quando os fluidos alcançam a superfície de um poço espontaneamente. Os poços que produzem através de elevação natural são conhecidos como poços surgentes. A espontaneidade na elevação dos fluidos nos poços surgentes dá-se pelo alto diferencial de pressão entre a pressão do reservatório e a pressão do poço, que faz com que o fluido saia do reservatório, local de maior pressão, e vá em direção à cabeça do poço, onde a pressão é menor.

A elevação natural ocorre, normalmente, no início da vida produtiva do reservatório, pois, à medida que o fluido é retirado, a pressão do reservatório tende a diminuir até se tornar insuficiente para emergir os fluidos numa vazão econômica ou conveniente (Thomas *et al.*, 2001).

Quando a elevação natural não é possível, torna-se necessária a aplicação de métodos artificiais de elevação. Os métodos de elevação artificial mais comuns na indústria do petróleo são:

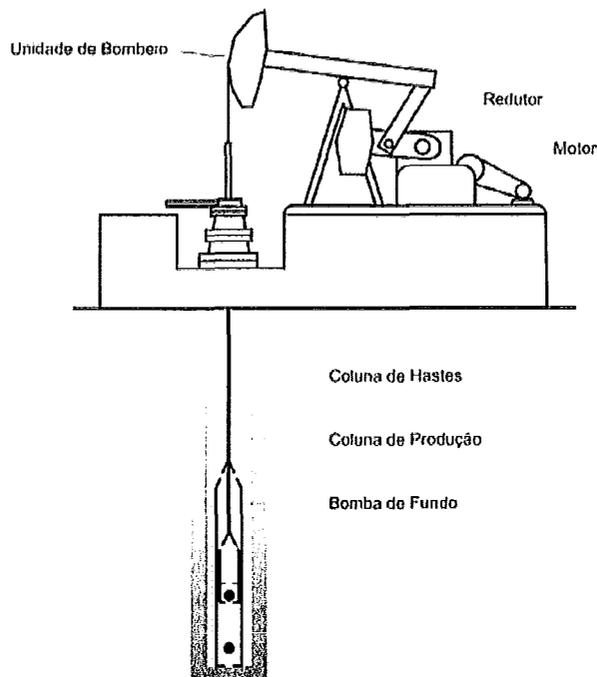
- Gás-Lift Contínuo (GLC);
- Gás-Lift Intermitente (GLI);
- Bombeio Centrífugo Submerso (BCS);
- Bombeio Mecânico com Hastes (BM);
- Bombeio por Cavidades Progressivas (BCP).

No próximo tópico, será descrito o sistema que opera através do bombeio mecânico com hastes (BM). Como o objeto de estudo desta dissertação é a identificação de falhas em bombeio mecânico, este será o único método de elevação apresentado neste trabalho.

## 2.1 Sistema de bombeio mecânico (SBM)

O bombeio mecânico é o método de elevação artificial de petróleo mais utilizado no Brasil (Barreto, 1993) e no mundo (Brown, 1982; Clegg e Buccaram, 1993; Ceylan, 2004). Segundo Costa (2005), sua popularidade está associada ao baixo custo de investimento e manutenção, grande flexibilidade de vazão e profundidade, eficiência energética e possibilidade de operação com fluidos de diferentes composições e viscosidades e em larga faixa de temperaturas. Brown (1982) cita, como uma das principais vantagens do bombeio mecânico, a sua facilidade de instalação e operação, sobretudo por funcionários que já estão familiarizados com esse método de elevação.

O sistema de bombeio mecânico (SBM) é aplicado somente para extração de óleo terrestre (*onshore*). Ele é composto, basicamente, pela unidade de bombeio (equipamento de superfície), a coluna de produção, a coluna de hastes e uma bomba de fundo.



**Figura 2.1.** Sistema de Bombeio Mecânico de Petróleo (Nascimento *et al.*, 2005).

A unidade de bombeio transforma o movimento rotativo do motor em um movimento alternativo. Transmitido para o fundo do poço através da coluna de hastes, o movimento alternativo aciona a bomba que eleva os fluidos produzidos pelo

reservatório até a superfície. A figura 2.1 mostra um sistema de bombeio mecânico simplificado.

Nesta seção, será apresentada a essência de cada um dos componentes principais do SBM. Em seguida, será explicado o conceito de carta dinamométrica de fundo. Por fim, serão apresentados os problemas mais comuns no bombeio mecânico que podem ser identificados a partir de uma carta dinamométrica.

## 2.1.1 Componentes Principais

### 2.1.1.1 Equipamentos de Superfície: A Unidade de Bombeio (UB)

Os equipamentos de superfície são os responsáveis pelo fornecimento e transmissão da energia no SBM. O conjunto de equipamentos de superfície são também conhecidos como Unidade de Bombeio (UB).

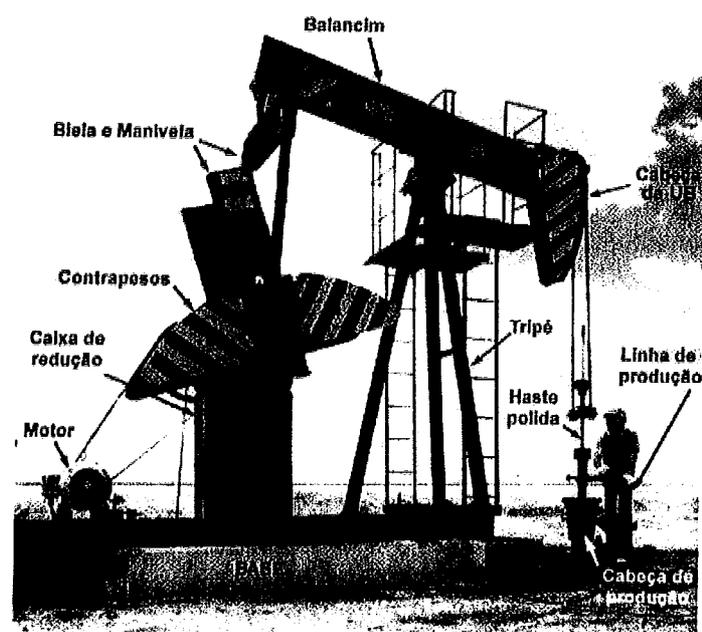


Figura 2.2. Equipamentos de Superfície ou Unidade de Bombeio (UB).<sup>1</sup>

Os componentes de superfície aparecem na figura 2.2. São eles: a base, o tripé, o motor, o redutor de velocidade (polia ou caixa de engrenagens), os contrapesos, a biela e a manivela, o balancim e a cabeça da UB.

<sup>1</sup> Foto de autoria de Eliana Fernandes, 1990 - segundo Thomas *et al.* (2001)

O motor, que pode ser elétrico ou de combustão interna, gera o movimento rotativo. A caixa de redução transforma a energia de alta velocidade e baixo torque do motor em energia de baixa velocidade e alto torque. Os contrapesos servem para reduzir o esforço no motor e no redutor. A biela e a manivela transmitem o movimento ao balancim. A posição onde a biela é presa à manivela determina o curso da haste polida. O balancim e a cabeça da UB transmitem o movimento à coluna de hastes e suportam toda a sua carga.

O objetivo dos equipamentos de superfície é, portanto, converter o movimento de rotação do motor da UB em um movimento alternativo – deslocamentos ascendente e descendente – da coluna de hastes.

A unidade de bombeio é caracterizada, basicamente, pelos seguintes parâmetros: geometria da sua estrutura, capacidade de carga dessa estrutura, curso máximo que pode efetuar e capacidade de torque do redutor.

### 2.1.1.2 A Coluna de Produção

A coluna de produção, também conhecida como coluna de tubos, é uma coluna composta por tubos de produção conectados através de luvas. A função da coluna de produção é conduzir os fluidos até a superfície e sustentar a bomba de fundo.

Para a produção em ambientes corrosivos ou em ambientes onde o atrito com as hastes é significativo, os tubos podem ser revestidos com fibra de vidro ou polietileno (*liners*), pinturas ou resinas.

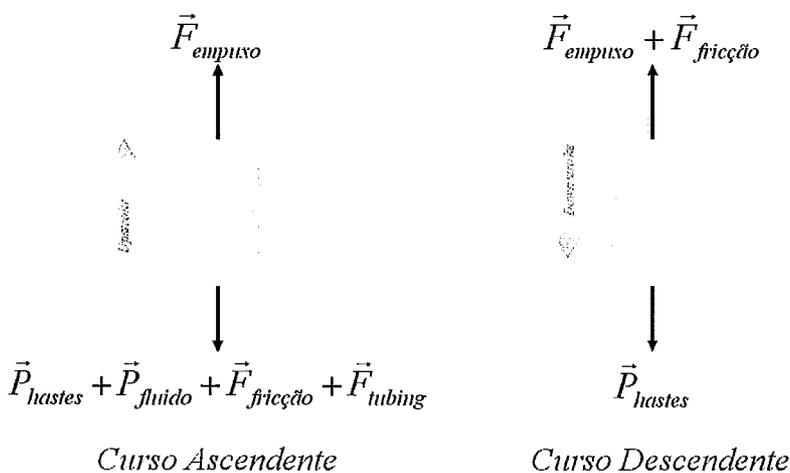
### 2.1.1.3 A Coluna de Hastes

A coluna de hastes faz a ligação entre o equipamento de superfície e a bomba de fundo. A haste polida se localiza no topo da coluna de hastes. Por sustentar todas as outras hastes, a haste polida está sujeita à maior carga exercida na coluna. As forças que atuam na haste polida são:

- Peso das hastes  $\vec{P}_{hastes}$  – Peso de toda a coluna de hastes;
- Peso do fluido  $\vec{P}_{fluido}$  – Peso da coluna de fluido que está acima do pistão;
- Força de empuxo  $\vec{F}_{empuxo}$  – Peso do fluido deslocado pelas hastes;

- Pressão do *tubing*  $\vec{F}_{tubing}$  – Força exercida pela coluna de produção (*tubing*), durante o curso ascendente, quando o fluido é deslocado dentro da coluna em direção à linha de produção.
- Força de aceleração  $\vec{F}_{aceleração}$  – Força responsável pela variação da velocidade das hastes;
- Força de fricção  $\vec{F}_{fricção}$  – Força é causada pelo atrito das hastes com o fluido e com a coluna de produção. No pistão, o atrito é causado pela presença de areia e parafina; nas hastes, pelo contato com a tubulação, com fluidos viscosos e também com parafina; e, na haste polida, pelo contato com as gaxetas.
- Cargas de choque – São cargas causadas quando ocorre pancada de fluido, batida de hastes ou *liners* desalinhados.

A carga total imposta à haste polida é a soma de todas as forças descritas acima. O peso das hastes é constante e atua no sentido da gravidade. A força de empuxo também é constante, mas atua de baixo para cima. A força de fricção é diretamente proporcional à velocidade das hastes e atua no sentido oposto ao do movimento da coluna. O peso do fluido e a pressão do *tubing* atuam somente no curso ascendente, quando a válvula de passeio está fechada. O funcionamento da válvula de passeio será explicado adiante.



**Figura 2.3.** Cargas estáticas e cargas de atrito que atuam na haste polida.

A Figura 2.3 mostra como essas forças atuam na haste polida. A aceleração, por ser uma carga dinâmica, muda de sentido durante o curso ascendente e durante o curso descendente: tem o mesmo sentido do movimento no início do curso, quando a velocidade das hastes está aumentando, e sentido contrário no fim do curso, a partir do momento em que começa a desaceleração. Então, para simplificação da figura, apenas as cargas de atrito e as cargas estáticas serão apresentadas.

A aceleração é máxima no *top stroke* (posição mais alta) e no *bottom stroke* (posição mais baixa), já que a velocidade é nula nesses pontos. Dessa forma, a carga na haste polida varia durante todo o ciclo de bombeio, mas é sempre positiva ou nula.

Além das cargas cíclicas a que estão expostas, as hastes trabalham em ambientes abrasivos e corrosivos. Por isso, a coluna de hastes é um ponto crítico do SBM.

#### 2.1.1.4 A Bomba de Fundo

A bomba de fundo consiste de um pistão que se desloca longitudinalmente no interior de uma camisa devido à energia transmitida pela coluna de hastes. A camisa da bomba nada mais é do que um barril suspenso na coluna de produção.

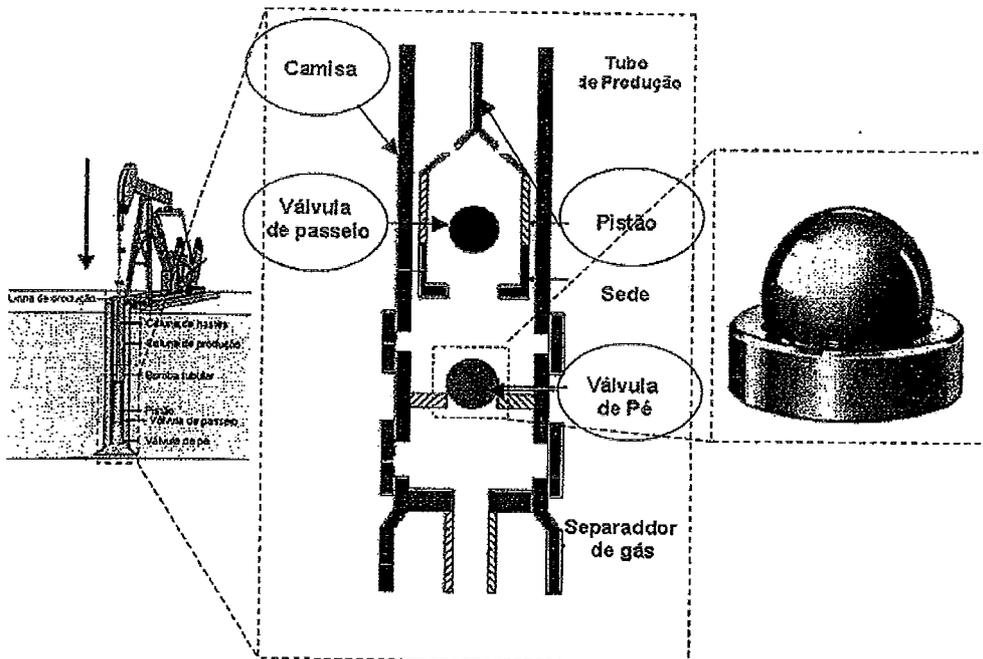
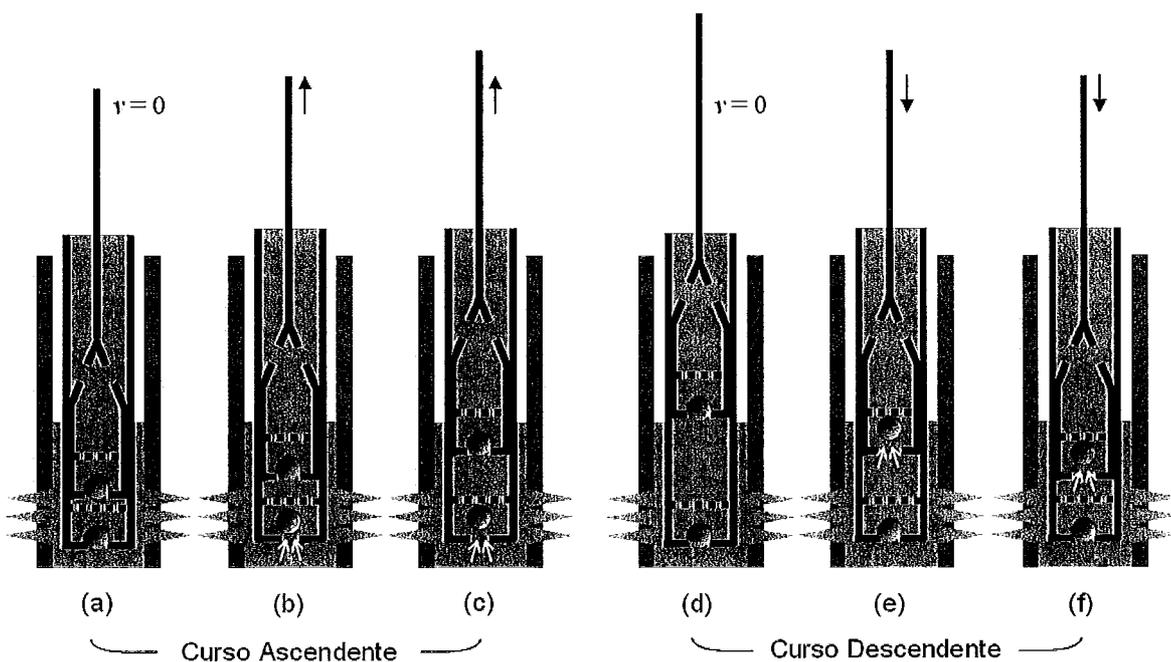


Figura 2.4. Bomba de fundo e seus principais elementos (Bezerra, 2002).

A bomba de fundo possui também duas válvulas cujo funcionamento é essencial para a eficiência do ciclo de bombeamento: a válvula de passeio e a válvula de pé. Essas

válvulas são do tipo sede e esfera e funcionam pela ação da diferença de pressão: a válvula se abre quando a pressão abaixo da esfera torna-se maior que a pressão acima; caso contrário, ela se fecha (Costa, 2005). A válvula de passeio pertence ao pistão e se move junto com ele. Já a válvula de pé fica alojada na parte inferior da camisa da bomba. A bomba de fundo e seus principais componentes – pistão, camisa, válvula de passeio e válvula de pé – podem ser vistos na Figura 2.4.

Cada vez que a coluna de hastes se desloca do fundo do poço – ponto morto inferior ou *bottom stroke* – até a cabeça do poço – ponto morto superior ou *top stroke* – e depois retorna ao fundo do poço, diz-se que um ciclo de bombeio foi concluído. Portanto, um ciclo de bombeio é formado por um movimento ascendente das hastes seguido de um movimento descendente. O movimento ascendente também é conhecido como *upstroke* e, o movimento descendente, como *downstroke*.



**Figura 2.5.** Curso ascendente e curso descendente do ciclo de bombeio. A Figura (a) representa o ponto morto inferior (*bottom stroke*), onde a velocidade das hastes é nula e, conseqüentemente, a aceleração é máxima. A partir desse ponto, inicia-se o curso ascendente, quando as hastes começam a se deslocar em direção à cabeça do poço. As figuras (b) e (c) mostram movimentos ascendentes. A figura (d) representa o ponto morto superior (*top stroke*) que tem o mesmo comportamento do ponto morto inferior: aceleração máxima devido à velocidade nula. Esse ponto marca o fim do curso ascendente e o início do curso descendente. As figuras (e) e (f) representam movimentos do curso descendente, que termina no *bottom stroke*, figura (a), dando início a um novo ciclo.

Quando a coluna de hastes se desloca em direção à cabeça do poço - curso ascendente ou *upstroke* -, a válvula de passeio mantém-se fechada pela pressão da coluna de fluido existente acima do pistão. Fechada a válvula de passeio, o fluido é elevado com as hastes dentro da coluna de produção em direção ao exterior do poço, isto é, para a linha de produção. Simultaneamente, a válvula de pé se abre para que o fluido escoe do reservatório para a camisa da bomba. Esse fluido somente será levado à superfície no próximo ciclo de bombeio. As figuras 2.5(a), 2.5(b) e 2.5(c) mostram uma seqüência de imagens de um curso ascendente.

Quando a coluna de hastes se desloca em direção ao fundo do poço - curso descendente ou *downstroke* -, a válvula de passeio é mantida aberta, permitindo a passagem do fluido à coluna de produção. Na verdade, a própria entrada do fluido faz com que a válvula de passeio permaneça aberta. Enquanto isso, a válvula de pé conserva-se fechada pela compressão de fluidos dentro da camisa, impedindo o retorno do fluido para o reservatório. As figuras 2.5(d), 2.5(e) e 2.5(f) mostram uma seqüência de imagens de um curso descendente.

### **2.1.2 Cartas Dinamométricas**

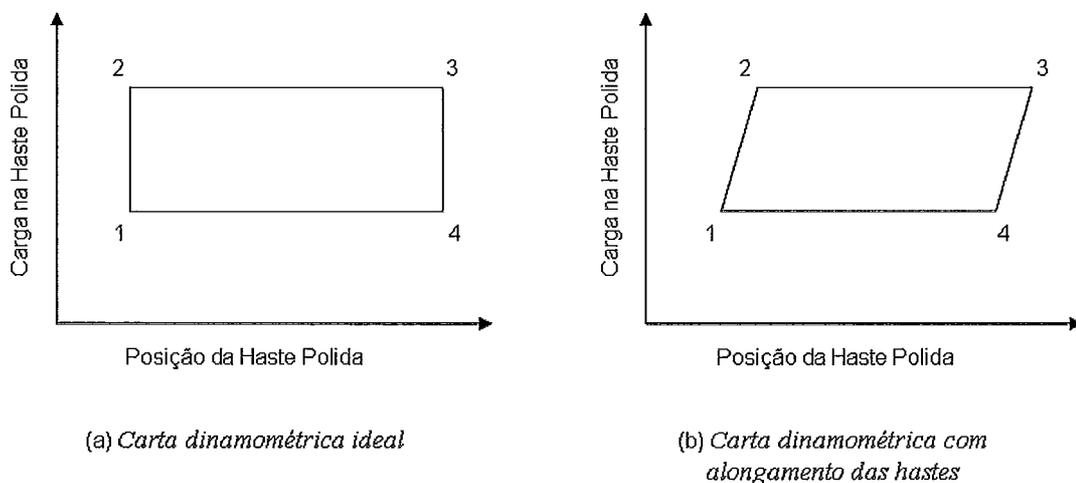
Um carta dinamométrica é um gráfico que mostra a variação de um conjunto de cargas em função do deslocamento da coluna de hastes. Há dois tipos de carta dinamométrica: a Carta Dinamométrica de Superfície (CDS) e a Carta Dinamométrica de Fundo (CDF). A diferença entre elas está no conjunto de cargas que cada uma apresenta: a CDS mostra as cargas que atuam na haste polida, enquanto a CDF exibe as cargas exercidas na bomba de fundo.

As cargas da CDS são medidas por um dinamômetro ou por sensores. As cargas da CDF também podem ser medidas por sensores, mas, devido ao ambiente altamente corrosivo a que a bomba está exposta, não é economicamente viável instalar e manter os sensores, fazendo-se necessários modelos matemáticos que calculem a CDF a partir da CDS.

A seguir, serão apresentados os conceitos de CDS e CDF detalhadamente, para que se possa entender a importância desse tipo de gráfico para uma rápida identificação de falhas no sistema de bombeio mecânico.

### 2.1.2.1 Carta Dinamométrica de Superfície (CDS)

A carta dinamométrica de superfície (CDS) é um gráfico que apresenta a variação da carga que atua na haste polida em função do deslocamento da coluna de hastes durante os cursos ascendente e descendente de um ciclo de bombeio. Um dinamômetro instalado na unidade de bombeio permite registrar as forças atuantes na extremidade superior da coluna de hastes durante o seu deslocamento.



**Figura 2.6.** Formatos típicos de uma CDS ancorada (a) e não-ancorada (b).

Takács (2002) afirma que uma CDS em condições ideais – isto é, em que a coluna de hastes é rígida e inelástica, a velocidade de bombeamento é baixa o suficiente para eliminar forças dinâmicas, o fluido bombeado é incompressível e todas as perdas de energia ao longo da coluna são desprezíveis – pode ser representada por um retângulo, como o que aparece na Figura 2.6(a).

O ponto 1 da Figura 2.6(a) representa o ponto morto inferior e o início do curso ascendente, mostrado também na Figura 2.5(a). No ponto 1, a carga atuante sobre a haste polida é apenas o peso da coluna no fluido ( $\vec{P}_{hastes} + \vec{F}_{empuxo}$ ). A esse peso, soma-se a carga do fluido, e a carga total –  $\vec{P}_{hastes} + \vec{F}_{empuxo} + \vec{P}_{fluido}$  – é transferida da válvula de pé imediatamente para a válvula de passeio quando esta se fecha, indicado pelo ponto 2. Do ponto 2 ao 3, quando as hastes se deslocam do ponto inferior ao ponto superior da coluna de produção, a carga permanece constante. No ponto 3, como foi mostrado na figura 2.5(d), o fluido acima do pistão é transferido para a linha de produção, a válvula de passeio se abre e a carga volta ao seu valor inicial, indicado pelo

ponto 4. As hastes se deslocam, então, em direção à posição inferior (ponto 4 ao ponto 1), também com carga constante. Ao retornar ao ponto 1, um novo ciclo começa.

A figura 2.6(b) mostra uma CDS em uma situação mais comum. Ela tem o formato de um paralelogramo, ao invés do retângulo, pois a transferência de carga da válvula de pé para a válvula de passeio (representada pela reta que liga o ponto 1 ao ponto 2) e a transferência de carga da válvula de passeio para a válvula de pé (representada pela reta que liga o ponto 3 ao ponto 4) ocorrem gradualmente em função do alongamento das hastes.

Os formatos das CDS apresentadas nas figuras 2.6(a) e 2.6(b) são, quase sempre, modificados devido às seguintes situações (Takács, 2002):

- Presença de cargas dinâmicas na haste geradas pela aceleração do movimento da coluna de hastes;
- A compressibilidade dos fluidos pode afetar extremamente a ação das válvulas de pé e passeio;
- Podem ocorrer problemas na subsuperfície que alterem a medição das cargas que atuam na haste polida;
- O movimento da haste polida e a operação da bomba de fundo induzem ondas de tensão na coluna de hastes. Essas ondas, transmitidas e refletidas na coluna de hastes, podem afetar consideravelmente a medição da carga na haste polida. Outra alteração significativa na medição poderá ocorrer se a frequência dessas ondas coincidir com a frequência fundamental da coluna de hastes.

Essas situações podem ocorrer isoladamente ou combinadas entre si. A figura 2.7 mostra as características de uma CDS real, considerando que a bomba de fundo está em perfeita condição mecânica e que não há interferência de gás.

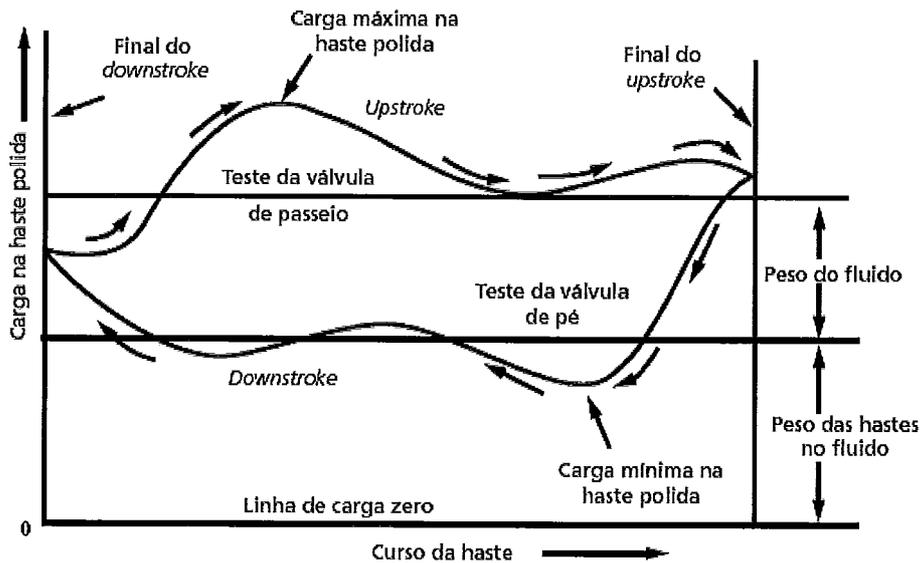


Figura 2.7. Carta Dinamométrica (Thomas *et al.*, 2001).

Gipson e Swaim (1984) apontaram seis cargas básicas que podem ser determinadas a partir de uma carta dinamométrica. Dentre elas, aparecem na Figura 2.7:

- Carga zero (*Base Line*) – Carga que servirá como referência para a medição de todas as outras cargas. Essa carga é obtida do dinamômetro sem cargas.
- Carga da válvula de pé (*SV Load*) – Representa o peso das hastes mergulhadas no fluido ( $\vec{P}_{hastes} + \vec{F}_{empuxo}$ ).
- Carga da válvula de passeio (*TV Load*) – Representa o peso das hastes mergulhadas no fluido somado ao peso do fluido sobre o pistão ( $\vec{P}_{hastes} + \vec{F}_{empuxo} + \vec{P}_{fluido}$ ).
- Carga máxima na haste polida (*Peak Polished Rod Load - PPRL*) – É a maior carga registrada durante o ciclo de bombeio. Equivale a carga da válvula de passeio somada à maior carga dinâmica sofrida no curso ascendente.
- Carga mínima na haste polida (*Minimum Polished Rod Load - MPRL*) – É a menor carga registrada durante o ciclo de bombeio. Equivale a carga da válvula de pé subtraída da maior carga dinâmica sofrida no curso descendente.

### 2.1.2.2 Carta Dinamométrica de Fundo (CDF)

A carta dinamométrica de superfície (CDS) é uma representação gráfica dos efeitos gerados pela carga atuante na bomba de fundo após terem se propagado pela coluna de hastes (Barreto Filho, 1993). O comportamento elástico da coluna de hastes durante a propagação é adicionado à força de tração medida, fazendo com que a CDS não represente o comportamento real da bomba de fundo, sobretudo em poços profundos.

Para resolver esse problema, Gibbs e Neely propuseram, em 1966, um método analítico<sup>2</sup> para calcular os pares de posição e carga atuante na bomba de fundo a partir dos pares de posição e força de tração na haste polida durante o ciclo de bombeio, isto é, um método para calcular a Carta Dinamométrica de Fundo (CDF) a partir da Carta Dinamométrica de Superfície (CDS). A partir de então, a carta dinamométrica de fundo tornou-se o principal recurso para se identificar problemas na operação de bombeio (Xu *et al.*, 2007).

#### Padrões de Carta Dinamométrica de Fundo

O formato de uma CDF representa a condição operacional do SBM. Isso significa que a carta pode representar tanto uma situação normal, quanto um problema no funcionamento do sistema. Nesta seção, serão apresentadas as formas características de cartas encontradas em diferentes situações.

#### **Normal**

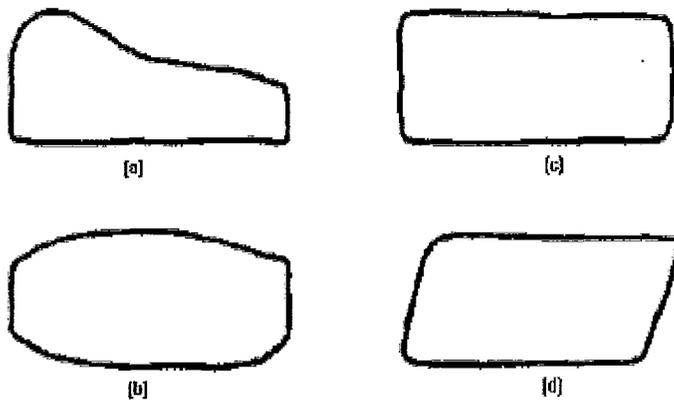
A carta normal representa o sistema de bombeio mecânico operando em uma condição esperada: alta eficiência volumétrica, pressão de sucção baixa ou média, e baixa interferência de gás.

Quando a coluna de hastes está ancorada, a carta normal apresenta o formato de um retângulo, como a figura 2.8(c). Caso contrário, quando a coluna não está ancorada ou quando há problemas na ancoragem, o efeito elástico da coluna faz com

---

<sup>2</sup>A solução desenvolvida por Gibbs e Neely está protegida pela patente U.S. 3,343,409 desde 1967 (Gibbs, 1967)

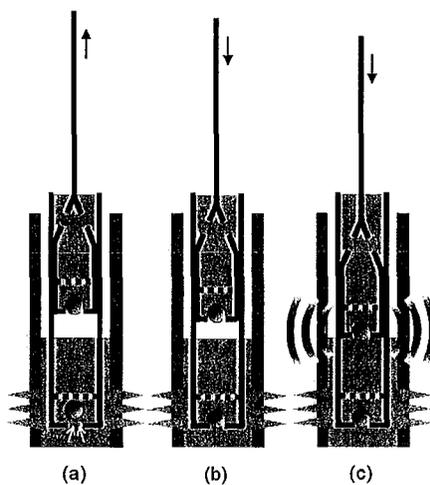
que a carta se pareça com um paralelogramo, como o da figura 2.8(d). Outras variações típicas no formato de uma carta normal são apresentadas também na figura 2.8.



**Figura 2.8.** Padrões de Carta Dinamométrica de Fundo Normal (Corrêa, 1998): (a) Normal com aceleração do Fluido; (b) Normal com atrito; (c) Normal Ancorada; (d) Normal Não-Ancorada.

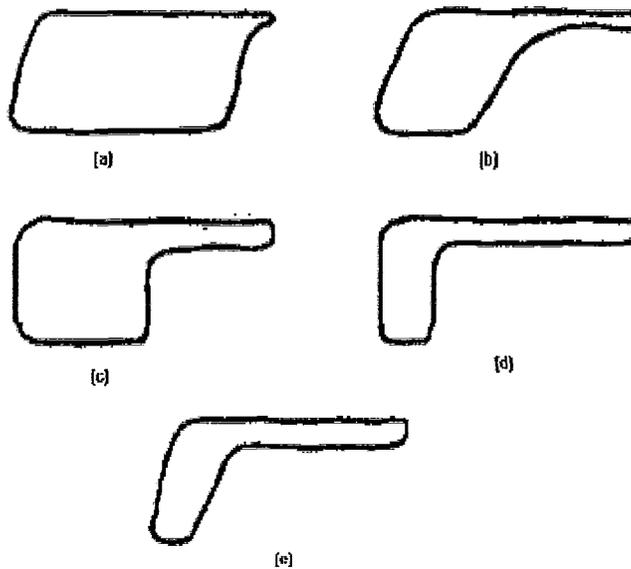
### Pancada de Fluido

A pancada de fluido representa o choque entre o pistão da bomba e o nível de óleo no interior da camisa no curso descendente. Geralmente, ocorre quando a bomba eleva uma quantidade de fluido superior àquela que flui do reservatório para o poço, de forma que a camisa da bomba não fique completamente cheia de fluido no curso ascendente, como mostra a Figura 2.9(a).



**Figura 2.9.** Pancada de Fluido.

Quando a camisa não está completa de fluido, a válvula de passeio não se abre no início do curso descendente (2.9.b), causando um forte impacto quando o pistão atinge o fluido (2.9.c). O impacto é transmitido à superfície ao longo da coluna de hastes.



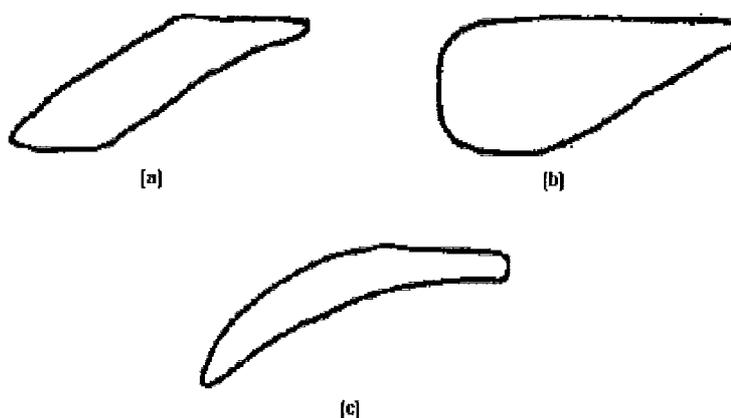
**Figura 2.10.** Padrões de Carta Dinamométrica de Fundo com Pancada de Fluido (Corrêa, 1998): (a) Pancada de fluido suave com coluna não-ancorada; (b) Pancada de fluido severa com coluna não-ancorada; (c) Pancada de fluido severa com coluna ancorada; (d) Pump-off completo (produção nula ou muito baixa) com coluna ancorada; (e) Pump-off completo com coluna não-ancorada.

Formatos característicos de uma CDF com pancada de fluido podem ser vistos na Figura 2.10.

### Interferência de Gás

Quando há grande quantidade de gás associado ao óleo, o choque entre o pistão e o nível do óleo é amortecido pelo gás no interior da camisa, fazendo com que a carga seja transferida da válvula de passeio para a válvula de pé suavemente durante o curso descendente, pois a válvula de passeio só se abre após a compressão suficiente do fluido.

Formatos típicos de uma CDF com interferência de gás são apresentados na Figura 2.11. Na interferência de gás, a ancoragem da coluna de tubos não influencia a forma da CDF significativamente.



**Figura 2.11.** Padrões de Carta Dinamométrica de Fundo com Interferência de Gás (Corrêa, 1998): (a) Interferência de gás severa; (b) Interferência de gás moderada; (c) Bloqueio de Gás.

Os choques que ocorrem na Pancada de Fluido e de Gás podem resultar em uma operação de bombeio ineficiente e até causar danos mecânicos no equipamento.

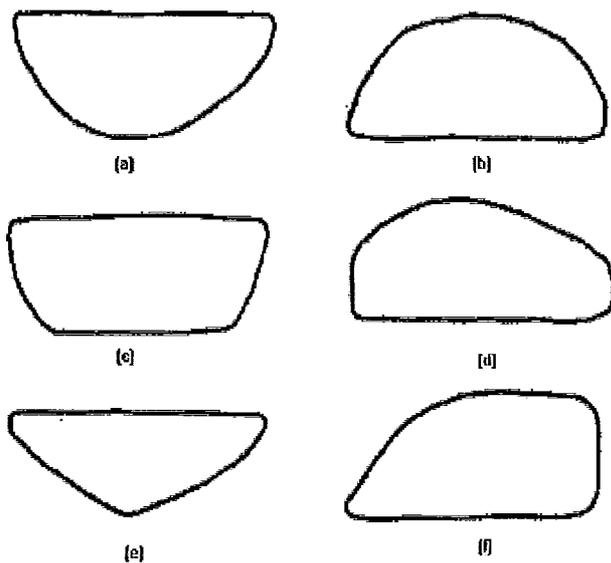
### Vazamento nas Válvulas

O vazamento nas válvulas de pé e de passeio podem ser causadas por desgaste na sede ou na esfera da válvula, por presença de sujeira ou parafina. Elementos abrasivos – como a areia – e corrosivos – como o sal, o  $H_2S$  e o  $CO_2$  – podem causar ou aumentar o desgaste das válvulas.

O vazamento na válvula de pé causa uma queda de pressão no curso descendente. Quando o vazamento ocorre na válvula de passeio, a queda de pressão aparece no curso ascendente. Em ambos os casos, há queda na produção de óleo e elevação do nível dinâmico.

O vazamento na válvula de pé causa a descarga prematura da carga sobre a válvula de passeio no curso ascendente e o atraso do retorno dessa carga à válvula de pé no curso descendente. A forma típica de uma carta com vazamento na válvula de pé aparece na Figura 2.12(a).

O vazamento na válvula de passeio faz com que, durante o curso ascendente, haja um atraso na transferência da carga do fluido sobre o pistão para a válvula de passeio e com que haja uma descarga prematura da carga de volta à válvula de pé, como pode ser visto na figura 2.12(b).

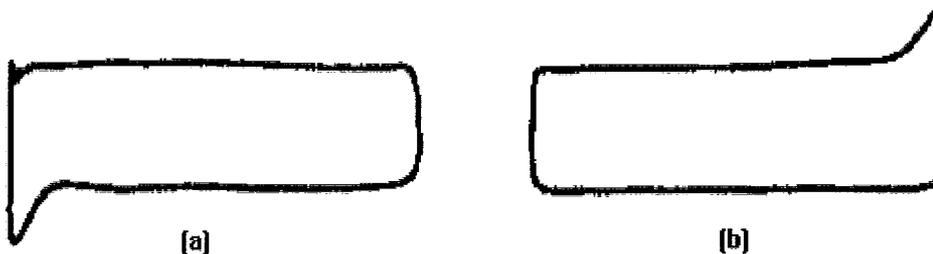


**Figura 2.12.** Padrões de Carta Dinamométrica de Fundo com Vazamento nas Válvulas (Corrêa, 1998): (a, e) Vazamento severo na válvula de pé; (b) Vazamento severo na válvula de passeio ou no pistão; (c) Vazamento moderado na válvula de pé; (d) Vazamento moderado na válvula de passeio ou no pistão; (f) Vazamento na válvula de passeio com coluna não-ancorada.

O vazamento na válvula de passeio é mais freqüente que o vazamento na válvula de pé.

### Batidas na Bomba

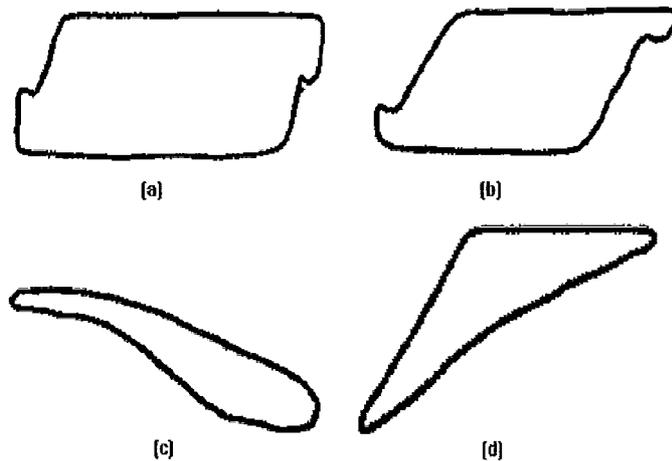
Batidas ocorrem somente em bombas insertáveis (Corrêa, 1998) e são causadas por espaçamento inadequado da bomba (Takács, 2002). A figura 2.13 (a) mostra uma batida no fundo da bomba e a figura 2.13 (b) apresenta uma batida no topo da bomba.



**Figura 2.13.** Padrões de Carta Dinamométrica de Fundo com Batidas na Bomba (Corrêa, 1998): (a) Batida no fundo da bomba; (b) Batida na parte superior da bomba.

## Problemas na Coluna ou no Pistão

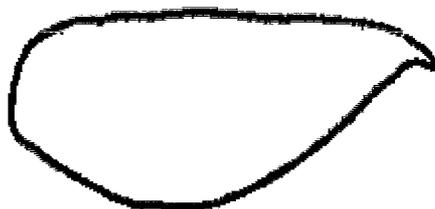
Outras condições de bombeio podem ser apresentadas por uma CDF. Dentre elas, estão algumas exibidas na figura 2.14: ancoragem inefficiente da coluna, pistão preso por areia e, até mesmo, ruptura da coluna de hastes. Nesses dois últimos casos, a produção é nula.



**Figura 2.14.** Padrões de Carta Dinamométrica de Fundo com outros problemas (Corrêa, 1998): (a, b) Ancoragem deficiente da coluna de hastes; (c) Ruptura da coluna de hastes (não há produção); (d) Pistão preso por areia.

## Efeitos sobrepostos

As condições apresentadas de operação de bombeio que podem ser determinadas pela CDF podem ocorrer isoladamente ou sobrepostas. Segundo Corrêa (1998), os efeitos superpostos na CDF mais comuns estão relacionados à interferência de gás e a problemas na bomba ou na coluna de hastes. A figura 2.15 mostra uma carta que apresenta vazamento nas válvulas e interferência de gás simultaneamente.



**Figura 2.15.** Exemplo de formato de CDF com superposição de efeito.

### 3 Redes Neurais Artificiais

O cérebro é um complexo sistema de processamento paralelo de informação. Embora computadores sejam extremamente rápidos e precisos na execução de instruções seqüenciais, eles são muito mais lentos que o cérebro humano para realizar o reconhecimento de sons, caracteres e imagens. A eficiência do cérebro em reconhecer padrões deve-se ao processamento altamente paralelo dos milhares de conexões de cada um de seus 100 bilhões de neurônios. As conexões entre os neurônios permitem ao cérebro humano adquirir habilidades para a execução de uma tarefa através de experiências, sem que seja necessário conhecer as regras que determinam tal tarefa.

Sistemas convencionais executam tarefas descritas através de algoritmos. Algumas vezes, porém, explicitar um conjunto de regras que rege uma atividade pode ser uma tarefa árdua ou, até mesmo, impossível. Nesses casos, torna-se essencial o aprendizado a partir de exemplos. Inspiradas na estrutura distribuída e maciçamente paralela do cérebro humano, as redes neurais artificiais vêm, portanto, suprir a lacuna deixada pelos sistemas convencionais em aprender a partir de um conjunto de padrões conhecidos e generalizar esse conhecimento ao se deparar com novos padrões.

Uma rede neural artificial é um sistema computacional altamente paralelo composto por simples unidades de processamento que imitam, de forma simplificada, o comportamento de neurônios biológicos e são capazes de armazenar conhecimento experimental e torná-lo disponível para o uso quando necessário.

Carvalho (2001) comenta em seu livro que o termo *Rede Neural Artificial* é uma tradução incorreta de *Artificial Neural Network*, posto que, em português, a palavra neural se refere ao sistema nervoso e não ao neurônio individualmente. Para evitar a tradução incorreta, a Sociedade Brasileira de Redes Neurais (SBRN)<sup>3</sup> optou pela adoção do termo *Rede Neural* – sem a palavra *Artificial* – para se referir a uma *Rede Neuronal Artificial*. Apesar disso, este trabalho adota o termo *Rede Neural Artificial* para se referir a redes compostas por neurônios, por esse termo ser o mais comum na literatura.

---

<sup>3</sup> <http://www.sbrn.org.br/>

### 3.1 Estrutura Básica

O neurônio biológico recebe sinais vindo de outros neurônios através de seus dendritos. Em seguida, ele combina os sinais de entrada e distribui a sua resposta a outros neurônios através de seu axônio. A figura 3.1 mostra a estrutura típica de um neurônio real.

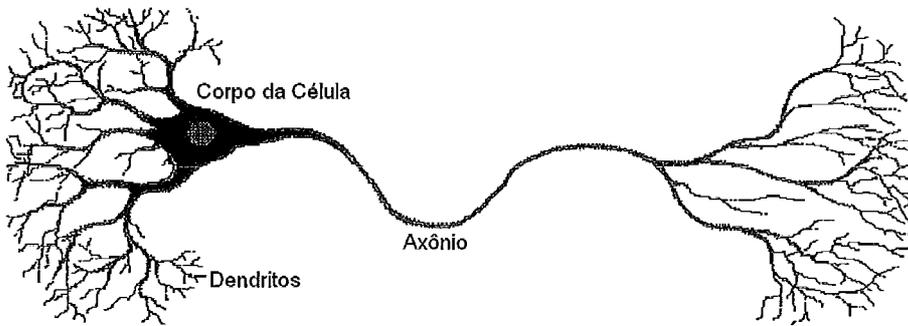


Figura 3.1. Neurônio Real (Reed e Marks, 1999).

De forma semelhante, o neurônio artificial, mostrado na figura 3.2, combina as entradas em um único valor que é transformado – através da função de ativação – para produzir a saída.

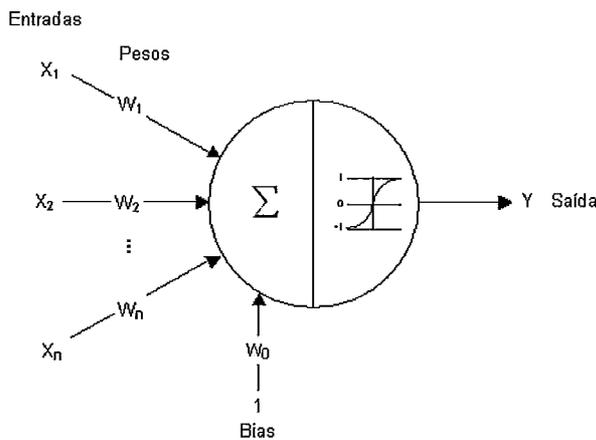


Figura 3.2. Neurônio Artificial<sup>4</sup>.

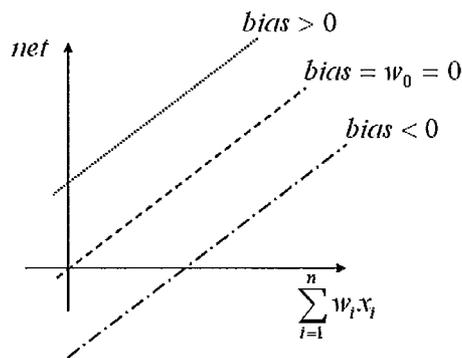
Assim, pode-se dizer que um neurônio artificial é composto, basicamente, por:

- Entradas – Correspondem aos dendritos de um neurônio biológico. Se o neurônio pertencer à primeira camada de processadores, as entradas

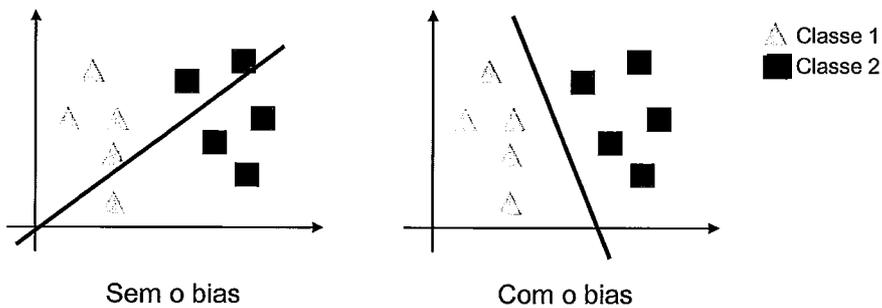
<sup>4</sup> Imagem baseada na figura 7.4 do livro de Berry e Linoff (2004).

corresponderão aos padrões apresentados à rede. Caso contrário, cada entrada será a saída de um neurônio da camada anterior.

- Pesos Sinápticos – Pesos associados às entradas e que representam a memória da rede neural. Cada peso determinará o efeito (inibição ou excitação) da entrada sobre o processador. Os pesos sinápticos representam a força de conexão entre os neurônios biológicos.
- O *bias* – termo polarizador – equivale a um peso associado a uma entrada com o valor fixo igual a 1. Ele define o domínio dos valores de saída, como pode ser visto na figuras 3.3.a e 3.3.b.
- Agregador ou Combinador Linear - combina todas as entradas ponderadas pelos seus respectivos pesos em um único valor:  $net = w_0 + \sum_{i=1}^n w_i x_i$
- Função de ativação - restringe a amplitude do intervalo de saída do neurônio, determinando o novo valor do estado de ativação do processador:  $y = f(net)$
- Saída – A saída do neurônio é calculada pela função de ativação. Corresponde ao axônio ou às sinapses de um neurônio biológico.



**Figura 3.3.a.** Influência do bias.



**Figura 3.3.b.** No primeiro gráfico, sem o bias, a reta passa pela origem e os dados não são separados corretamente. No segundo, com o bias, a reta deslocada da origem fornece uma melhor classificação para os dados.

Com a estrutura descrita acima, cada neurônio é capaz de adquirir conhecimento através de um processo de aprendizagem e armazená-lo nos pesos sinápticos. O processo de aprendizagem é definido por um algoritmo que modifica os pesos da rede até alcançar o objetivo proposto.

A modificação dos pesos sinápticos permite a mutação da topologia da rede, tornando-a mais apta para determinada aplicação. Se, por exemplo, após o processo de aprendizagem, uma rede apresentar todos os pesos que chegam a um determinado neurônio como sendo nulos, significa que esse neurônio nunca será ativado e, portanto, não é necessário na rede. Dessa forma, o comportamento de uma rede neural é determinado pelo modo como os neurônios estão conectados (topologia da rede) e pelos valores dessas conexões (os pesos sinápticos).

As principais áreas de atuação das redes neurais artificiais são a classificação de padrões, categorização (*clustering*), previsão de séries temporais, otimização, reconhecimento de padrões e aproximação de funções.

### 3.1.1 Funções de Ativação

O objetivo da função de ativação é limitar a amplitude do intervalo de saída do neurônio. As funções de ativação mais comuns são baseadas no modelo biológico onde a saída permanece muito baixa até que as entradas alcancem um valor mínimo (*threshold*). A partir desse valor, a unidade é ativada e a saída torna-se alta (Berry e Linoff, 2004). Alguns exemplos de funções de ativação podem ser vistas na figura 3.4.

As funções lineares são, normalmente, escolhidas para os neurônios da camada de saída quando se deseja que a resposta da rede neural seja apenas a combinação linear da saída de cada neurônio da camada anterior. Uma rede composta somente de neurônios com função de ativação linear fará o papel de uma regressão linear. Portanto, as funções de ativação mais interessantes são as funções não-lineares.

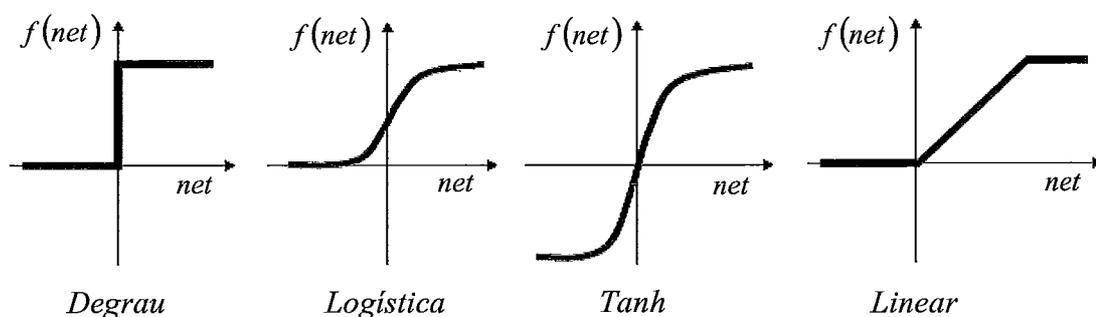


Figura 3.4. Funções de Ativação.

A função logística fornece saídas entre 0 e 1 e, a função tangente hiperbólica, valores entre -1 e 1, mas ambas apresentam um comportamento linear quando a soma ponderada de suas entradas apresenta um valor próximo a 0.

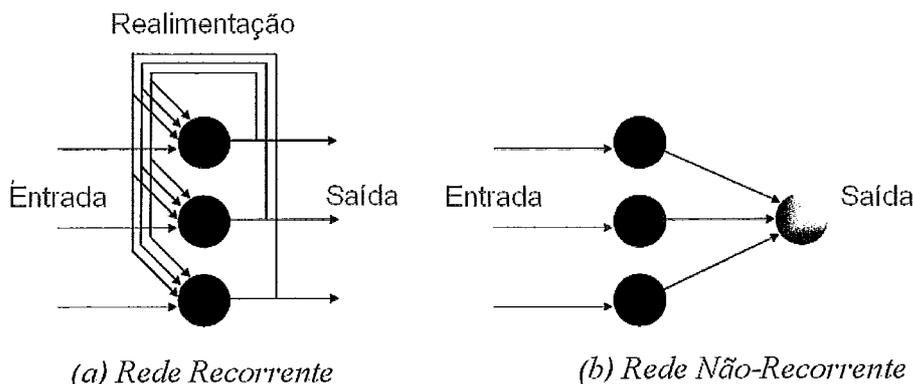
Segundo Bishop (1995), aplicadas transformações lineares adequadas na entrada e na saída de uma função tangente hiperbólica, ela se torna equivalente a uma função logística e, por isso, não há diferença entre uma rede cujos neurônios da camada escondida usem uma função hiperbólica e outra rede em que usem uma sigmóide, exceto pelos diferentes valores que terão para os seus pesos e biases. Apesar disso, empiricamente, as funções hiperbólicas tendem a convergir mais rápido que as logísticas durante o treinamento da rede.

## 3.2 Arquitetura de uma Rede Neural

A arquitetura de uma rede neural é definida por sua topologia, isto é, pela forma como os seus processadores – os seus neurônios – estão interconectados. Basicamente, existem dois tipos de topologia para interconexão dos processadores: as redes recorrentes e as redes não-recorrentes.

As redes recorrentes são redes com conexões entre processadores da mesma camada e/ou com processadores das camadas anteriores. Por isso, diz-se que as redes

recorrentes são redes com realimentação. A figura 3.5(a) mostra um exemplo de rede recorrente em que os neurônios recebem, além do vetor de entradas, a saída de todos os neurônios daquela camada.



**Figura 3.5.** Topologias de redes neurais.

As redes não-recorrentes são redes de uma ou mais camadas de processadores cujo fluxo de dados é sempre em uma única direção, ou seja, não possuem realimentação de suas saídas para as suas entradas e, por isso, são ditas “sem memória”. Um exemplo desse tipo de arquitetura são as redes do tipo *Feed-Foward*, em que o sinal é sempre propagado para a frente, da entrada para a saída, como a que aparece na figura 3.5(b).

As redes não-recorrentes podem ser formadas por uma ou várias camadas de neurônios. A maioria dos autores não considera as entradas como uma camada da rede porque não elas realizam o processamento típico de um neurônio.

Nas redes formadas por uma camada única, somente uma camada de pesos separa as entradas das saídas. Nas redes multicamadas, há uma camada de saída e uma ou mais camadas intermediárias. As camadas intermediárias são chamadas também de camadas escondidas ou ocultas. Essa denominação vem da tradução do termo inglês *hidden layers*.

### 3.3 Aprendizado de uma Rede Neural

A rede neural adquire conhecimento através de um processo de aprendizagem iterativo que, em geral, consiste de repetidas apresentações de um conjunto de dados à rede, acompanhadas do ajustamento dos seus pesos. A seguir serão explicadas as formas

de apresentação desses dados e, em seguida, os algoritmos básicos para ajuste dos pesos da rede.

### 3.3.1 Forma de apresentação dos dados: Batch x On-line

O conjunto de dados utilizado para o aprendizado da rede é denominado de Conjunto de Treinamento. Cada apresentação desse conjunto à rede é chamada de ciclo (ou época) de treinamento. Se o ajuste dos pesos for realizado ao fim de cada ciclo, ou seja, após a apresentação de todo o conjunto de treinamento à rede, tem-se um Aprendizado por Ciclos (*Batch Learning*). Caso contrário, se o ajuste dos pesos ocorrer após a apresentação de cada padrão, diz-se que se trata de um Aprendizado Por Padrão ou Padrão-a-Padrão (*On-Line Learning* ou *Pattern-Mode Learning*).

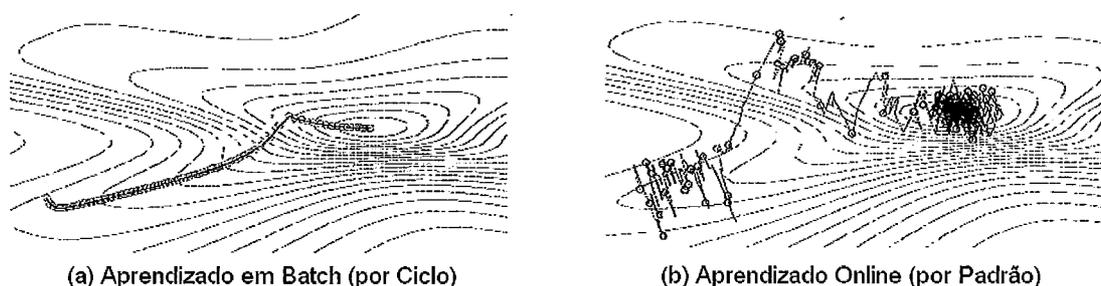


Figura 3.6. Tipos de Treinamento. Figuras de Reed e Marks (1999).

No aprendizado em *Batch*, cada padrão é avaliado com a mesma configuração dos pesos a cada ciclo e, portanto, não importa a ordem em que os padrões são apresentados à rede. No aprendizado *On-line*, como os pesos são ajustados após a avaliação de cada padrão, a rede tende a aprender melhor o último padrão apresentado. Para evitar efeitos cíclicos, Reed e Marks (1999) sugerem que os padrões sejam apresentados à rede em ordem aleatória e constantemente alterada quando se optar pelo aprendizado incremental.

A apresentação randômica dos padrões necessária no aprendizado *On-line* elimina o comportamento suave do erro – como o que é apresentado pelo aprendizado em *Batch* na figura 3.6(a) –, e passa a ter descidas e subidas bruscas, como as que aparecem na figura 3.6(b).

Embora o aprendizado incremental tenha menos chance de ficar preso em um mínimo local, ele necessita de uma frequência maior de atualização dos pesos, o que pode torná-lo mais lento que o aprendizado por ciclos. Além disso, acredita-se que o ajuste dos pesos após serem apresentados todos padrões, como ocorre no aprendizado em *batch*, seja mais preciso do que se realizado a cada padrão.

### **3.3.2 Tipos de treinamento**

O treinamento é um processo incremental de ajuste dos pesos de uma rede neural para atingir um determinado objetivo. Os tipos básicos de treinamento são:

- Treinamento Supervisionado;
- Treinamento Não-Supervisionado;

#### **3.3.2.1 Treinamento Supervisionado**

O objetivo de um treinamento supervisionado é tornar a rede apta a fornecer as saídas esperadas quando apresentado um conjunto de entradas. Para isso, os seus pesos são ajustados a fim de se minimizar o erro obtido ao se comparar a saída da rede com a saída desejada.

O treinamento supervisionado torna a rede um aproximador de função, onde os pesos correspondem aos coeficientes de um polinômio e as entradas correspondem aos valores que as variáveis do polinômio podem assumir. Assim, o ajuste dos pesos é nada mais que o ajuste dos coeficientes de uma função, até que ela retorne o valor esperado.

As Redes Multilayer Perceptron (MLP), que serão apresentadas na seção 3.3.4, são exemplos de rede que utilizam o treinamento supervisionado.

A figura 3.7 apresenta o funcionamento do aprendizado supervisionado.

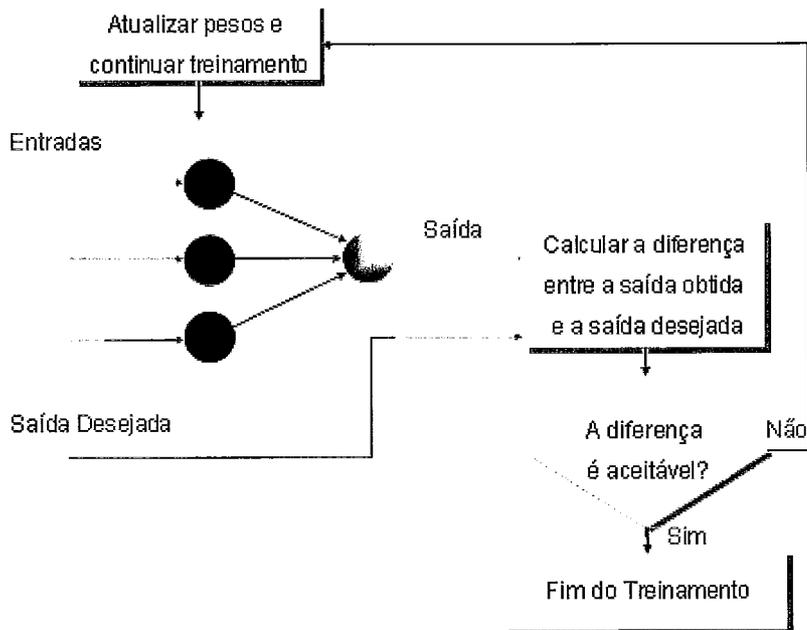


Figura 3.7. Treinamento supervisionado.

### 3.3.2.2 Treinamento Não-Supervisionado

Normalmente, o treinamento não-supervisionado é realizado quando não se conhece a saída desejada para cada padrão que será apresentado à rede. Por isso, esse tipo de treinamento é muito aplicado em problemas de agrupamento (ou, em inglês, *clustering*), quando se tem um conjunto de padrões, mas não se sabe como eles estão relacionados.

O treinamento não-supervisionado permite que os dados sejam distribuídos em classes de acordo com as características extraídas do conjunto de padrões. A figura 3.8 mostra um exemplo do funcionamento do treinamento não-supervisionado. Nesse exemplo, cada padrão é representado por duas entradas:  $x_1$  e  $x_2$ . A classe A e a classe B não são conhecidas no começo do treinamento. Os pesos, com valores inicialmente aleatórios, são representados pelos vetores  $W_A$  e  $W_B$ . Se um padrão apresentado à rede estiver mais próximo do vetor de peso da classe A, então  $W_A$  irá se mover em direção a esse padrão. O mesmo ocorre se o padrão estiver mais próximo à classe B:  $W_B$  se move em direção ao padrão. Dessa forma, à medida que os padrões vão sendo apresentados,  $W_A$  e  $W_B$  vão se aproximando do centro da classe que representam.

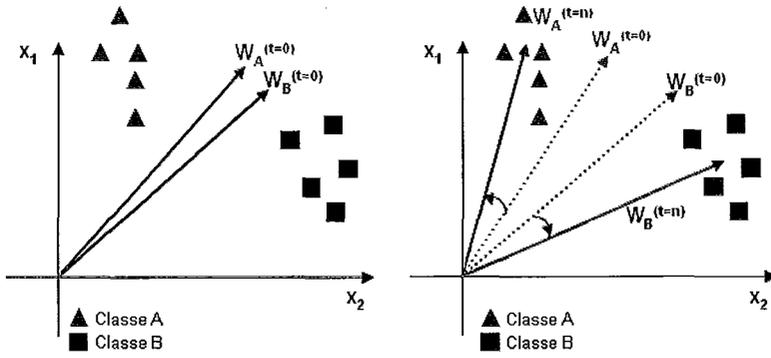


Figura 3.8. Exemplo de um treinamento não-supervisionado.

Mapas Auto-Organizáveis (*Self-Organizing Maps*) são exemplos de redes com treinamento não-supervisionado. Esse tipo de rede será visto em detalhes na seção 3.3.7.

### 3.3.3 Redes Perceptron de uma camada

O termo *Perceptron* é normalmente usado para se referir a redes com uma única camada de neurônios que têm a função degrau como função de ativação. A função gerada por Redes Perceptron de uma camada representará sempre um hiperplano (definido por  $w_0 + \sum_{i=1}^n w_i x_i$ ), onde os padrões ficarão de um lado do hiperplano se a saída for positiva e, de outro, se for negativa.

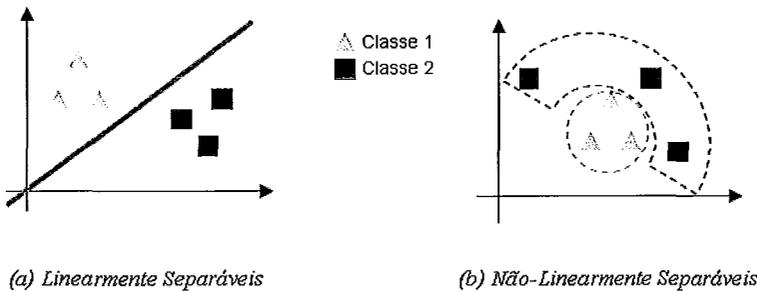
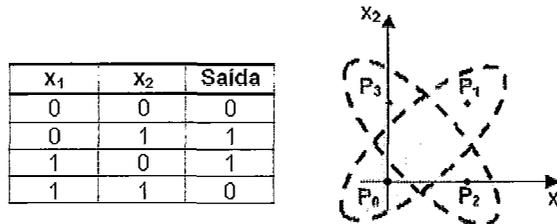


Figura 3.9. Na Figura a, tem-se um exemplo de duas classes que podem ser separadas por um hiperplano (como só há duas entradas, o hiperplano será uma reta). Na Figura b, há duas classes que não podem ser separadas por um hiperplano e, portanto, não são linearmente separáveis.

Com uma camada, apenas a inclinação e a posição do hiperplano poderão ser modificadas em função da alteração dos pesos e do bias. Por isso, redes Perceptron de

uma camada são restritas a representar funções linearmente separáveis, nas quais se pode separar classes distintas por um hiperplano, como ocorre na Figura 3.9(a).

Um exemplo de função não-linearmente separável é a função XOR (ou-exclusivo). A figura 3.10 mostra que, com as possíveis entradas da função XOR –  $P_0 = (0,0)$ ;  $P_1 = (1,1)$ ;  $P_2 = (1,0)$ ; e  $P_3 = (0,1)$  –, não é possível definir um hiperplano que separe os pontos  $P_0$  e  $P_1$  (que têm saída igual a 0) em regiões diferentes de  $P_2$  e  $P_3$  (que têm saída igual a 1).



**Figura 3.10.** Representação da função XOR.

Minsky e Papert (1969) mostraram então que, para resolver problemas não-linearmente separáveis, é necessário uma rede multicamadas. As redes Multilayer Perceptron serão descritas a seguir.

### 3.3.4 Redes Multilayer Perceptron (MLP)

Redes Multilayer Perceptron (MLP) são redes com mais de uma camada de neurônios, isto é, redes com, pelo menos, uma camada escondida além da camada de saída. Nesse tipo de rede, cada neurônio é conectado a todos os neurônios da camada seguinte.

A figura 3.11 mostra um exemplo de rede MPL com três camadas de neurônios: duas camadas intermediárias (camadas escondidas) e uma camada de saída. Essa rede recebe 5 variáveis como entrada e fornece 2 saídas.

As redes de multicamadas surgiram após ser constatado que redes de apenas uma camada só representam funções linearmente separáveis. Geralmente, as redes MLP adotam o algoritmo de treinamento conhecido como *Backpropagation*.

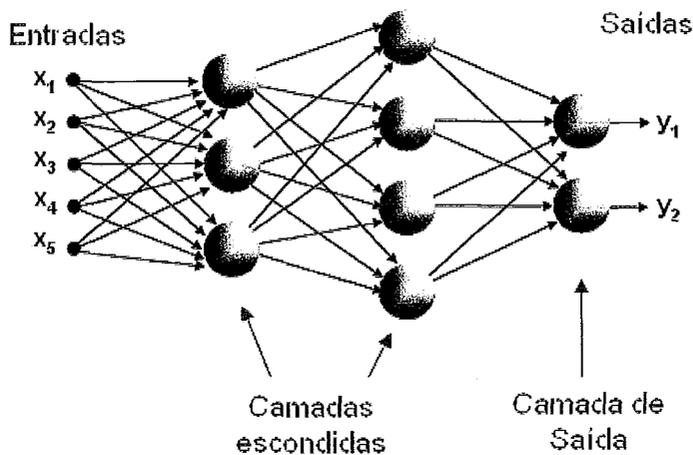


Figura 3.11. Exemplo de uma rede MLP de 3 camadas: duas camadas escondidas e uma camada de saída.

### 3.3.4.1 *Backpropagation* (Retropropagação)

*Backpropagation* é o algoritmo mais usado para treinamento de redes multicamadas do tipo *Feed-forward*. O nome do algoritmo vem do fato dos erros gerados na saída serem retropropagados para as camadas intermediárias até que todos os pesos sejam atualizados.

O algoritmo mais comum de *Backpropagation* é conhecido como Gradiente Descendente. O peso de cada entrada de um neurônio é atualizado proporcionalmente ao negativo da derivada parcial do erro da saída desse neurônio.

Nesse algoritmo, dois parâmetros influenciam o aprendizado da rede neural: a *taxa de aprendizado* e o *momento*. A *taxa de aprendizado* controla o quão rápido os pesos mudam: quanto maior for o valor da taxa, mais rápido será o aprendizado da rede neural. Apesar disso, a *taxa de aprendizado* não pode ser muito grande porque a rede pode se tornar instável. A taxa também não pode ser pequena, pois, se o algoritmo estiver preso em um mínimo local, uma mudança nos pesos pode fazer o erro aumentar, impedindo que consiga sair do mínimo local. Para evitar oscilações e que o algoritmo fique preso em um mínimo local, a abordagem recomendada é começar com um valor alto para a taxa e ir diminuindo-o à medida que a rede vai sendo treinada.

O termo *momento* determina a proporção da última mudança de peso que é adicionada à nova mudança de peso. A inclusão do *momento* permite à rede convergir mais rapidamente, pois penaliza a troca de sinais da derivada da função de ativação,

acelerando a busca no espaço dos pesos para um erro que seja mínimo global e, ao mesmo tempo, evitando as oscilações na rede durante o treinamento.

Outro algoritmo de treinamento por retropropagação de erros muito utilizado é o Levenberg-Marquardt. O algoritmo Levenberg-Marquardt é o aperfeiçoamento do método Gauss-Newton, que é uma variante do método de Newton. O método de Newton usa a informação da derivada parcial de segunda ordem da função de performance utilizada para corrigir os pesos da rede. Portanto, o método Levenberg-Marquardt usa, além da informação do gradiente, a informação sobre a curvatura da superfície do erro (Ranganathan, 2004).

Na iniciação do processo iterativo, o método de Levenberg-Marquardt se comporta de forma semelhante ao Gradiente descendente. À medida que o processo iterativo se aproxima da solução do problema, o método de Levenberg-Marquardt se assemelha ao método de Gauss-Newton.

A principal desvantagem do método de Levenberg-Marquardt é a grande quantidade de memória que ele precisa para armazenar algumas matrizes durante a sua execução. O tamanho da matriz que guarda a primeira derivada, por exemplo, é  $Q \times n$ , onde  $Q$  é o número de amostras do conjunto de treinamento e  $n$  é o número de pesos e biases da rede neural. Por isso, esse algoritmo é indicado para treinamento de redes de tamanho moderado, com poucas centenas de pesos (Demuth, Beale e Hagan, 2008).

### 3.3.5 Capacidade das redes MLP

Em 1998, Cybenko demonstrou que uma rede MLP com duas camadas escondidas permite a aproximação de qualquer função e são suficientes para representar regiões de qualquer tipo. Em 1989, Cybenko demonstrou que as redes do tipo MLP que apresentam apenas uma camada intermediária são aproximadores universais para funções contínuas.

Duas camadas de perceptrons - uma camada intermediária e uma camada de saída - podem, portanto, formar qualquer região convexa, mas, para representar regiões complexas, é necessário utilizar duas camadas escondidas, além da camada de saída (Lippmann, 1988).

### 3.3.5.1 Convergência x Generalização

A capacidade de generalização de uma rede neural está associada à sua capacidade de responder corretamente a padrões desconhecidos, isto é, fornecer as saídas esperadas para entradas que não foram utilizadas durante o treinamento. A convergência, por outro lado, é a capacidade da rede de responder corretamente a padrões já vistos durante o aprendizado. Há, então, um dilema entre a convergência e a generalização: quando se ganha em um, perde-se em outro. A seguir, serão abordadas algumas questões sobre esse conflito.

#### Tamanho da rede

Os pesos representam a memória da rede neural. Se o número de neurônios for menor que o necessário, não haverá pesos suficientes para aprender todos os padrões do conjunto de treinamento. Nesse caso, perde-se a convergência.

Por outro lado, se o número de neurônios de uma rede neural for muito grande, haverá um número excessivo de pesos, fazendo com que a rede modele, além dos aspectos fundamentais dos dados, a contribuição dos ruídos. Ao capturar os detalhes específicos de um conjunto de treinamento particular, a rede terá perdido a sua capacidade de generalização, pois terá “decorado” os exemplos desse conjunto e não responderá corretamente a padrões nunca vistos. Dessa forma, a rede acertará a saída de todos os padrões que pertencerem ao conjunto de treinamento, mas não será capaz de generalizar o conhecimento adquirido no treinamento para lidar com ruídos e distorções de novos padrões.

#### Número de Ciclos de Treinamento

O método mais utilizado para se determinar o número de ciclos de treinamento, obtendo-se uma boa generalização da rede, é conhecido como Parada Antecipada (*Early Stopping*). O método da Parada Antecipada consiste em separar um conjunto de validação – conjunto de padrões não utilizados para atualização dos pesos – e apresentá-lo à rede para identificar o ponto a partir do qual ocorre o supertreinamento (*overfitting*), isto é, ciclo a partir do qual o erro dos padrões do conjunto de validação começa a subir de forma consistente.

A figura 3.12 mostra a performance de uma rede durante os ciclos (épocas) do treinamento. Como a atualização dos pesos ocorre apenas ao final de cada ciclo, o erro do treinamento (curva azul) decresce durante todo o aprendizado da rede. O erro de validação decresce até o 3º ciclo. No 4º ciclo, ele aumenta, mas volta a cair na próxima época. Somente a partir do 5º ciclo, o erro de validação começa a aumentar de forma consistente, mostrando que a rede começa a decorar e a perder sua capacidade de generalização. Para se ter uma boa generalização da rede no exemplo dado, o treinamento deveria ser interrompido no 5º ciclo.

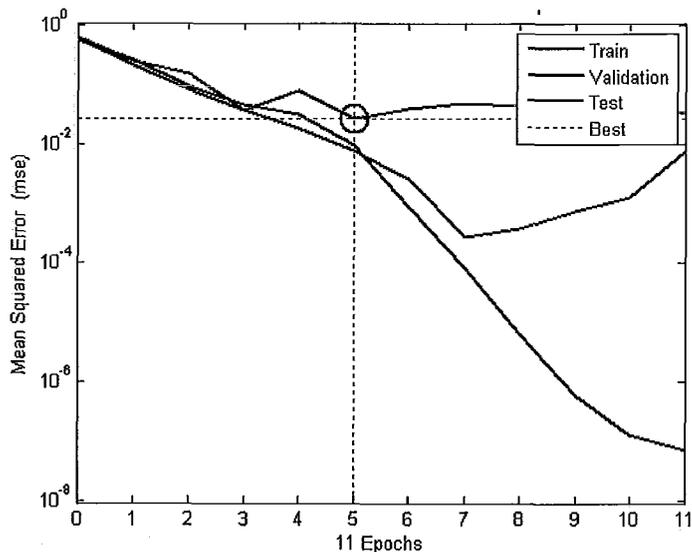
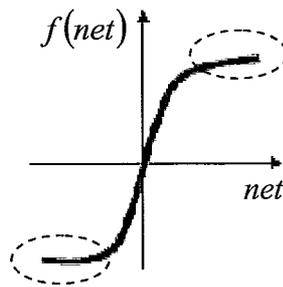


Figura 3.12. Gráfico que mostra a performance de uma rede para os conjuntos de treinamento, validação e teste. O gráfico foi extraído a partir do software MATLAB.

### 3.3.5.2 Paralisia da rede

Quando a soma das entradas de um neurônio ponderadas pelo peso de cada conexão  $\left( net = w_0 + \sum_{i=1}^n w_i x_i \right)$  assume um valor alto (em módulo), ao passar pela função de ativação, a saída atinge as regiões onde a derivada da função  $f(net)$  é aproximadamente nula, como mostra a Figura 3.13. Isso faz com que o ajuste dos pesos  $\Delta w$  também se aproxime de zero e o aprendizado se torne cada vez mais lento.



**Figura 3.13.** Função de ativação, destacando-se as regiões onde a derivada aproxima-se de zero.

Redes neurais que têm pesos iniciais e/ou variáveis de entrada com valores muito altos são causas típicas de saturação da rede. As soluções comuns são iniciar os pesos com pequenos valores randômicos e normalizar as entradas de forma que fiquem em um intervalo razoável (Reed e Marks, 1999).

A redução do número de neurônios nas camadas escondidas diminui o número de pesos da rede e, assim, também ajuda a evitar o problema de paralisia da rede.

### 3.3.6 Comparação entre modelos

Conforme explicado anteriormente, redes com diferentes arquiteturas serão treinadas com os padrões do conjunto de treinamento. Um conjunto de validação é necessário para evitar o supertreinamento da rede. Para testar o desempenho final do modelo, é necessário submeter um conjunto de teste à rede. A amostra de padrões deve, portanto, ser dividida em três conjuntos:

- **Conjunto de Treinamento (Estimação)** – contém os padrões usados para modificar os pesos. Em geral, o conjunto de treinamento tem, aproximadamente, 70% dos padrões;
- **Conjunto de Validação** - contém os padrões usados para verificar o problema de *overfitting* (supertreinamento). Costuma-se usar cerca de 20% dos padrões;
- **Conjunto de Teste** - contém os padrões usados para testar o desempenho do modelo final. Normalmente, o conjunto de teste contém cerca de 10% dos padrões.

Se a amostra de padrões for pequena, o conjunto de treinamento selecionado poderá não ser o mais adequado para fornecer o melhor desempenho na generalização. Então, para que o erro obtido no conjunto de testes represente, de fato, o erro que será obtido durante o uso real do modelo, é necessário que os conjuntos representem toda a população de dados. Segundo Witten e Frank (2005), não há como identificar se um conjunto é representativo ou não, mas deve-se garantir, pelo menos, que cada classe presente no conjunto total seja representada com a mesma proporção no conjunto de treinamento, de validação e de teste.

Para tentar se obter o conjunto de treinamento que forneça o melhor desempenho na generalização, pode-se realizar a validação cruzada dos dados. Será explicada a seguir, de forma simplificada, uma forma de validação cruzada conhecida como *K-fold cross-validation*.

### 3.3.6.1 K-Fold Cross-Validation

A execução da validação cruzada K-fold cross-validation pode ser representada pelos seguintes passos:

1. Escolher um número fixo  $K$  de partições do conjunto total de dados;
2. Dividir o conjunto total em  $K$  partições;
3. Escolher uma das  $K$  partições, ainda não escolhida, para ser o conjunto de teste (também conhecido como *holdout*). As  $K - 1$  partições serão divididas no conjunto de treinamento e no conjunto de validação;
4. Avaliar o erro do conjunto de teste definido pela partição escolhida;
5. Se todas as partições já foram escolhidas para ser o conjunto de teste, calcular a média de todos os  $K$  erros encontrados. Senão, voltar ao *passo 3*.

Segundo Witten e Frank (2005), a validação cruzada com 10 partições (*10-Fold Cross-Validation*) é o método padrão para se medir a performance (taxa de erro) de um modelo de aprendizado sobre um determinado conjunto de dados. Devido à iniciação aleatória dos pesos da rede, aconselha-se que a validação cruzada seja rodada, pelo menos, 10 vezes para se ter resultados confiáveis.

### 3.3.6.2 Métricas para comparação de modelos

O erro de uma avaliação mede o quanto as saídas diferem dos valores esperados. Na área de estatística, há diversas métricas para se quantificar o erro de um modelo preditivo. Dentre elas, estão:

- *MSE - Mean Squared Error*

$$MSE = \frac{1}{n} \sum_{i=1}^n (Y_i - y_i)^2$$

Onde:

- $n$  é o número de padrões apresentados;
- $Y_i$  é o valor esperado para o  $i$  –ésimo padrão;
- $y_i$  é a saída fornecida pelo modelo para o  $i$  –ésimo padrão;

Assim,  $Y_i - y_i$ , a diferença entre o valor esperado  $Y_i$  e o valor encontrado  $y_i$ , representa o erro do  $i$  –ésimo padrão.

- *RMSE - Root Mean Squared Error*

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (Y_i - y_i)^2}$$

- *MAPE - Mean Absolute Percentage Error*

$$MAPE = \frac{1}{n} \sum_{i=1}^n \left| \frac{Y_i - y_i}{Y_i} \right| \times 100\%$$

Onde:

- $\frac{Y_i - y_i}{Y_i}$  representa o erro proporcional ao valor esperado.

No MAPE, a diferença entre o valor esperado  $Y_i$  e o valor encontrado  $y_i$  é dividida pelo valor esperado para se ter um erro proporcional a esse valor.

A tabela 3.1 apresenta uma comparação entre o erro absoluto e o erro proporcional de um conjunto de par de valores obtido e esperado.

$Y_i$	$y_i$	$ Y_i - y_i $	$\left  \frac{Y_i - y_i}{Y_i} \right $
10,000	5,000	5,000	0,500
1005,000	1000,000	5,000	0,005
1005,000	5,000	1000,000	0,995
0,001	5,000	4,999	4999,000

**Tabela 3.1.** Erro Absoluto x Erro Proporcional.

O erro proporcional parece fazer mais sentido, pois um erro absoluto não nos diz nada sem ser comparado com outro valor. Por exemplo, o erro absoluto igual a 5, calculado nas duas primeiras linhas, representa um erro grande em relação ao valor esperado 10, porém, em relação ao valor 1005, representa um erro bem pequeno.

É importante notar que métricas que usam o erro proporcional não devem ser usadas quando se tem valores esperados próximos a zero, pois o erro poderá assumir valores absurdos, como mostra a última linha da tabela.

- *U - Theil's U*

$$U = \frac{\sqrt{\sum_{i=1}^n (Y_i - y_i)^2}}{\sqrt{\sum_{i=1}^n (Y_i - Y_{i-1})^2}}$$

Onde:

- $Y_{i-1}$  representa o valor esperado para o último padrão avaliado.

A métrica Theil's U, diferente das demais, mede o quão bom é o modelo comparado a um modelo ingênuo que, simplesmente, repete o valor esperado do último padrão avaliado. Por isso, essa métrica é, normalmente, utilizada para avaliar modelos de previsão de séries temporais.

Quanto mais U se aproxima de 0, melhor é a performance do modelo preditivo. Se U for igual a 1, o modelo não oferece ganho, pois tem performance equivalente a um modelo que apenas fornece a última observação como valor previsto.

### 3.3.7 Redes Não-Supervisionadas - Competitive Learning

Como já foi mencionado, o treinamento não-supervisionado é uma forma de aprendizado que divide o conjunto de padrões de entrada em grupos (clusters) inerentes aos dados, quando não existe conhecimento a respeito da classe a que o padrão pertence.

Como não se tem um vetor de resposta desejado para efetuar comparações com as saídas obtidas, a rede se ajusta às regularidades estatísticas dos dados de entrada. Por isso, esse tipo de aprendizado necessita de um grande conjunto de dados, pois precisa de redundância para adquirir conhecimento das propriedades estatísticas dos padrões.

As redes não-supervisionadas são baseadas em aprendizado competitivo (*Competitive Learning*). Em sua forma mais simples, o aprendizado competitivo usa apenas uma camada de neurônios, onde cada neurônio representa um grupo. Ao apresentar um padrão à rede, o neurônio mais próximo ao vetor de entrada será o vencedor, e os pesos desse neurônio serão ajustados para que ele se aproxime do padrão de entrada que o ativou. Assim, vetores de entrada similares devem ativar o mesmo neurônio.

A atualização dos pesos minimiza a diferença entre o vetor de pesos e o vetor de entrada do neurônio vencedor, como foi visto na figura 3.8(b). O algoritmo em que apenas os pesos do neurônio vencedor são ajustados é conhecido como *Winner Takes All*.

Como o número de classes (ou grupos) não é conhecido a priori, pode ser que um neurônio represente mais de uma classe (criando um supergrupo) ou que uma classe seja criada por mais de um neurônio (criando um subgrupo). Para resolver esse problema, surgiram os Mapas Auto-Organizáveis de Kohonen.

#### 3.3.7.1 *Self-Organizing Maps (SOM)* ou Mapas de Kohonen

Nos mapas auto-organizáveis, os neurônios são distribuídos em um arranjo bidimensional. A idéia principal é que cada classe possa ser representada por mais de um neurônio. Os neurônios que representam uma classe devem possuir localizações geográficas semelhantes no mapa. Assim, padrões similares devem ativar neurônios próximos um do outro no mapa.

Os mapas auto-organizáveis de Kohonen são baseados no aprendizado competitivo e, por isso, seguem os mesmos princípios: os neurônios da camada de saída competem entre si e um, com o valor mais alto, “vence”. Porém, nos mapas de Kohonen, não se atualizam somente os pesos que conduzem ao neurônio vencedor, ajustam-se também os pesos dos neurônios vizinhos ao neurônio vencedor. Dessa forma, o neurônio vencedor e os seus vizinhos ficarão fortalecidos para o padrão de entrada apresentado. O mecanismo de vizinhança dos mapas de Kohonen resolve o problema dos subgrupos e supergrupos.

O ajuste dos pesos do mapa é controlado pela taxa de aprendizado e por um parâmetro que indica o tamanho da vizinhança. O tamanho da vizinhança e a taxa de aprendizado são grandes no início e decrescem com o passar do tempo de treinamento.

### O SOM Toolbox

O SOM Toolbox<sup>5</sup> é um pacote de funções para o Matlab<sup>6</sup>, onde se encontra a implementação do algoritmo de Kohonen. A base de dados esperada pelo SOM Toolbox é um arquivo em formato texto com colunas separadas por espaço, onde cada coluna representa uma variável e, cada linha, uma amostra. O layout geral do arquivo pode ser visto na figura 3.14.

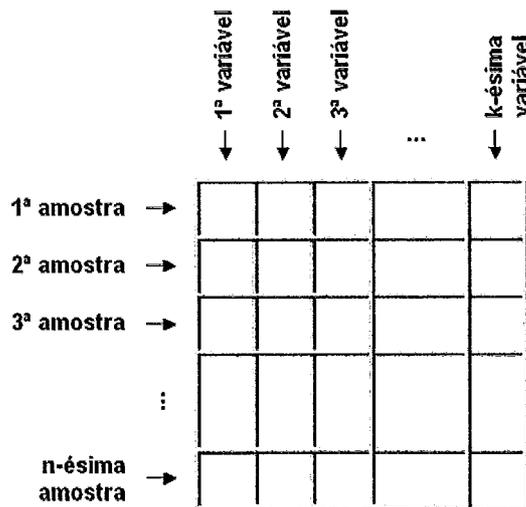


Figura 3.14. Layout do arquivo de entrada do SOM Toolbox.

<sup>5</sup> <http://www.cis.hut.fi/projects/somtoolbox/>

<sup>6</sup> <http://www.mathworks.com/>

A figura 3.15 mostra um exemplo<sup>7</sup> de um arquivo de entrada do SOM Toolbox. A primeira linha do arquivo indica o número de variáveis e a segunda linha informa os nomes das variáveis, que devem ser sempre precedidos por “#n”. A última coluna é opcional e representa o rótulo (*label*) da linha. Esse rótulo ajudará na avaliação do agrupamento sugerido pelo Mapa de Kohonen.

#n	p1	c1	p2	c2	p3	c3	...	p100	c100	
200	0.01	0.06	0.00	0.07	0.00	0.08	...	0.01	0.06	Normal
	0.01	0.06	0.00	0.07	0.00	0.08	...	0.01	0.06	Normal
	0.00	0.05	0.00	0.06	0.00	0.07	...	0.00	0.05	Normal
	0.01	0.07	0.00	0.07	0.00	0.08	...	0.01	0.07	Normal
	0.00	0.06	0.00	0.07	0.00	0.07	...	0.00	0.06	Normal
	0.00	0.05	0.00	0.06	0.00	0.06	...	0.00	0.05	Normal
	0.00	0.06	0.00	0.07	0.00	0.08	...	0.00	0.06	Normal
	0.01	0.06	0.00	0.07	0.00	0.08	...	0.01	0.06	Normal
	0.01	0.07	0.00	0.08	0.00	0.08	...	0.01	0.07	Normal
	0.01	0.10	0.01	0.09	0.01	0.09	...	0.01	0.10	Normal
	0.00	0.02	0.00	0.06	0.00	0.12	...	0.00	0.02	Pancada
	0.01	0.01	0.00	0.05	0.00	0.12	...	0.01	0.01	Pancada
	0.01	0.01	0.00	0.05	0.00	0.12	...	0.01	0.01	Pancada
	0.00	0.02	0.00	0.05	0.00	0.12	...	0.00	0.02	Pancada
	0.00	0.02	0.00	0.05	0.00	0.12	...	0.00	0.02	Pancada
	0.00	0.02	0.00	0.05	0.00	0.12	...	0.00	0.02	Pancada
	0.00	0.01	0.00	0.05	0.00	0.12	...	0.00	0.01	Pancada
	0.00	0.03	0.00	0.07	0.00	0.13	...	0.00	0.03	Pancada
	0.00	0.03	0.00	0.07	0.00	0.13	...	0.00	0.03	Pancada
	0.00	0.03	0.00	0.07	0.00	0.14	...	0.00	0.03	Pancada

Figura 3.15. Exemplo de arquivo de entrada do SOM Toolbox.

A figura 3.16 mostra um exemplo de uso do SOM Toolbox com o arquivo da figura 3.15 completo. O resultado desse código é uma matriz – a matriz U – que representa o mapa de Kohonen com uma célula adicionada entre cada par de neurônios para mostrar a distância entre eles. A cor da célula entre o par de neurônios se aproxima da cor vermelha se a distância entre os neurônios for grande e, da azul, se for pequena.

A figura 3.17 mostra a matriz U e o mapa de Kohonen ao lado, destacando-se os neurônios que foram mais ativados através dos rótulos dos padrões que os ativaram. A matriz U apresenta uma linha de células vermelhas que separa o mapa em duas regiões principais. Pelo mapa que contém os rótulos, pode-se ver que os padrões apresentados à rede foram agrupados corretamente em suas respectivas classes, identificadas pelos rótulos “Normal” e “Pancada”, apresentadas na parte inferior e superior do mapa, respectivamente.

<sup>7</sup> Na imagem 3.15, que representa um exemplo de arquivo de entrada do SOM Toolbox, as colunas foram separadas por “TAB”, ao invés de por espaço, apenas para efeito de visualização.

```

% Lê o arquivo de dados e carrega-os na matriz D
D = som_read_data('cartas.dat');

% Normaliza os dados.
D = som_normalize(D,'range');

% Cria a rede (7x3) e inicializa os pesos
M = som_randinit(D,'msize', [7 3]);

% Treina a rede
M = som_batchtrain(M,D,'radius',[4 1],'trainlen', 300);

% Exibe a matriz U e o mapa com os neurônios mais ativados rotulados
som_show(M,'umat','all','empty','Labels','norm','d');
M = som_autolabel(M,D,'vote');
som_show_add('label',M, 'subplot',2);

```

Figura 3.16. Exemplo de código usando o SOM Toolbox.

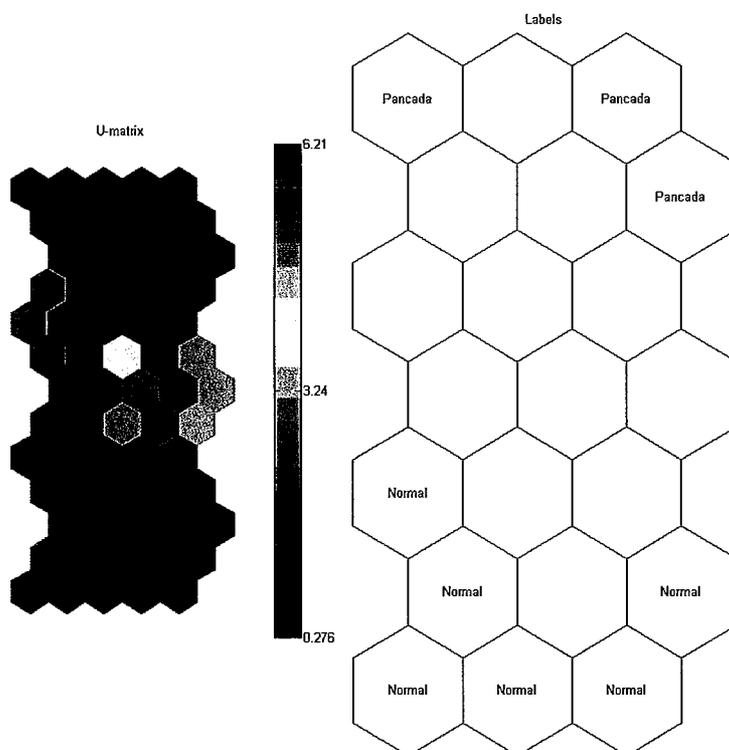


Figura 3.17. Resultado do código da figura 3.16.

### Métricas para Avaliação da Qualidade do Mapa Treinado

Há duas métricas para se medir a qualidade do mapa após o processo de aprendizagem com o conjunto de treinamento: o erro médio de quantização e o erro topológico.

O erro médio de quantização  $e_{quant}$  é a distância média entre cada vetor de dados e o vetor de pesos do neurônio vencedor:

$$e_{quant} = \frac{\sum_{i=1}^n |X_i - W_{ij}|}{n}$$

Onde:

- $X_i$  é o vetor do  $i$ -ésimo padrão apresentado;
- $W_{ij}$  é o vetor de pesos que liga o  $i$ -ésimo padrão ao neurônio vencedor  $j$ . Esse vetor também é conhecido como BMU (*Best Match Unit*).

O erro topológico  $e_{topol}$  é definido pela percentagem de vetores de dados para os quais o primeiro e o segundo BMUs não são unidades (neurônios) adjacentes no mapa:

$$e_{topol} = \frac{\sum_{i=1}^n u(x_i)}{n}$$

Onde:

- $u(x_i) = \begin{cases} 1, & \text{se 1º BMU e 2º BMU não são adjacentes} \\ 0, & \text{caso contrário} \end{cases}$

O erro topológico permite avaliar o quanto a estrutura do mapa aproxima padrões próximos no espaço de entrada (Silva, 2004).

### 3.3.8 Redes ART

As redes ART (*Adaptive Resonance Theory* ou, em português, Teoria da Ressonância Adaptativa) são arquiteturas para aprendizagem não-supervisionada propostas por Gail Carpenter e Steve Grossberg. Essas redes classificam padrões compostos por valores binários, onde cada um desses valores representa a presença ou a ausência de uma das características usadas como entrada.

Há duas características importantes em redes não-supervisionadas: a estabilidade e a plasticidade. A estabilidade representa a capacidade da rede de associar

um padrão que não foi classificado a um grupo já existente. A plasticidade mede a capacidade da rede de criar uma nova classe para um padrão que não se assemelhe aos grupos existentes. Segundo Carvalho (2001), a Teoria da Ressonância Adaptativa (ART) foi desenvolvida para detectar semelhanças entre padrões, mantendo a estabilidade e, ao mesmo tempo, a plasticidade.

Quando há semelhança entre o padrão de entrada e um dos neurônios – o neurônio vencedor –, diz-se que a rede neural entrou em ressonância. Nesse caso, ocorre o aprendizado da rede: o neurônio vencedor se aproxima do padrão que o ativou, da mesma forma como ocorrem nos demais tipos de rede de aprendizado competitivo.

Na arquitetura ART, um parâmetro de vigilância  $\rho$  é utilizado para verificar a semelhança entre o padrão de entrada e os neurônios da camada de saída. Segundo Nepomuceno (2003), quando  $\rho$  assume um valor baixo (inferior a 0,4), a rede permite que padrões não muito semelhantes sejam agrupados na mesma classe, mas, se  $\rho$  assumir um valor alto, pequenas diferenças entre os padrões de entrada levarão a criação de novas classes

### 3.3.8.1 Redes Fuzzy-ART

As Redes Fuzzy-ART permitem a classificação de padrões compostos por valores entre 0 e 1. Nesse caso, cada valor representa a pertinência ao conjunto fuzzy de uma das características que compõe a entrada. A figura 3.18 compara a lógica ART binária com a ART analógica (lógica fuzzy).

ART (BINÁRIO)	FUZZY-ART (ANALÓGICO)
<b>ESCOLHA DE CATEGORIAS</b>	
$T_j = \frac{ I \cap w_j }{\alpha +  w_j }$	$T_j = \frac{ I \wedge w_j }{\alpha +  w_j }$
<b>CRITÉRIOS DE SIMILARIDADE</b>	
$\frac{ I \cap w }{ I } \geq \rho$	$\frac{ I \wedge w }{ I } \geq \rho$
<b>APRENDIZADO RÁPIDO</b>	
$w_j^{novo} = I \cap w_j^{velho}$	$w_j^{novo} = I \wedge w_j^{velho}$
$\cap$ = operador lógico E (Intersecção)	$\wedge$ = operador nebuloso E (Mínimo)

Figura 3.18. Analogia entre as redes ART e Fuzzy-ART (Nepomuceno, 2003).

A iniciação dos pesos da rede Fuzzy-ART não é aleatória: todos os pesos começam com o valor 1. O aprendizado sempre converge porque os pesos vão decrescendo à medida que a rede vai se adaptando aos (aprendendo os) padrões de treinamento. Mas, se muitos pesos convergirem a zero, pode ocorrer uma proliferação de categorias. Esse problema pode ser evitado preservando-se a amplitude de cada característica que compõe a entrada da rede. Por isso, além da normalização dos padrões de entrada, a rede Fuzzy-ART faz a codificação por complemento de cada padrão. Assim, o vetor de entradas  $\{a\}$  terá seu tamanho dobrado (figura 3.19), pois será representado também pelo seu complemento  $\{a^c\}$ , onde  $a^c = 1 - a$ , e, ao invés de se utilizar o vetor de entradas  $\{a\}$ , será utilizado o vetor  $\{a, a^c\}$ .

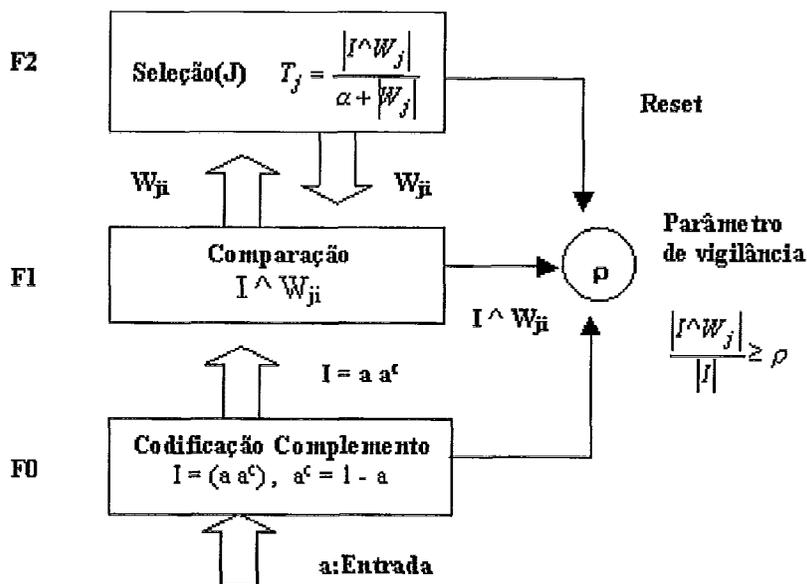


Figura 3.19. Arquitetura Fuzzy-ART. (Cámara e Oliveira, 2003).

### 3.3.8.2 Redes Fuzzy ARTMAP

As Redes Fuzzy ARTMAP são redes com treinamento supervisionado que adotam uma regra de aprendizado que minimiza o erro de predição e, ao mesmo tempo, maximiza a generalização (Carpenter e Grossberg, 1992). Isso é alcançado pelo processo - conhecido como *Match Tracking* - que incrementa o parâmetro de vigilância ART pela quantia mínima necessária para corrigir o erro preditivo. Dessa forma, o sistema descobre o menor número de categorias – número de neurônios na camada escondida – que fornece os resultados esperados.

Diferente da maioria das arquiteturas de redes, o Fuzzy ARTMAP trabalha com aprendizado *On-line* – aprende cada padrão à medida que o recebe –, o que faz com que a ordem em que os padrões são apresentados à rede faça diferença no resultado do treinamento. Assim, simulações com o conjunto de treinamento em diferentes ordenações levam a redes com diferentes pesos e diferentes quantidades de neurônios na camada escondida. Por isso, o ARTMAP Fuzzy utiliza uma estratégia de votação: realizam-se diversas simulações com os padrões em diferentes ordens e a previsão final é aquela que ocorrer na maioria das simulações.

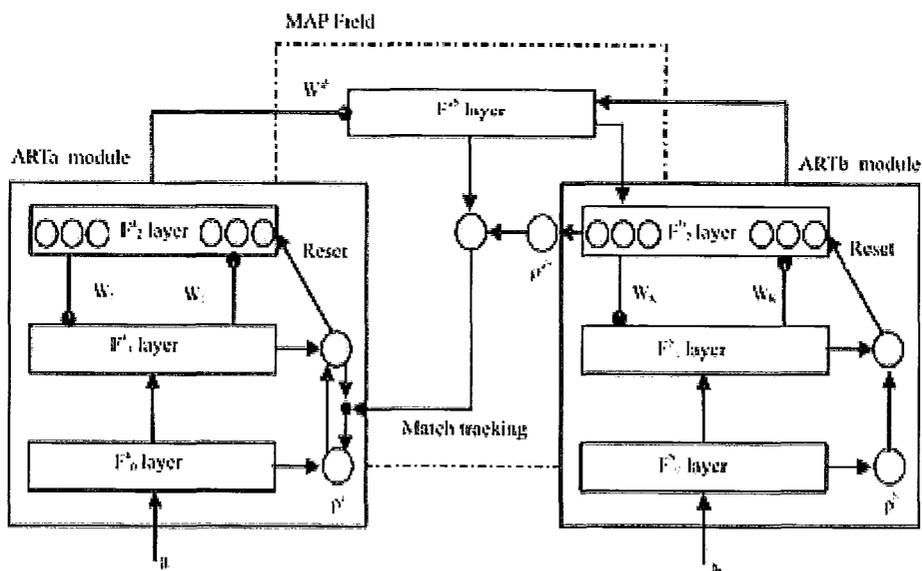


Figura 3.20. Arquitetura Fuzzy ARTMAP. (Alilat *et al.*, 2006).

A figura 3.20 mostra a arquitetura da rede Fuzzy ARTMAP. O módulo  $ART_a$  recebe o vetor  $\{a\}$  composto pelas entradas da rede (*inputs*). O módulo  $ART_b$  recebe o vetor  $\{b\}$ , que são as saídas esperadas (*targets*). A camada  $F_0$  de cada módulo encontra o complemento de cada vetor recebido e repassa-os – o vetor e o seu complemento – à camada superior  $F_1$ . Um vetor de pesos adaptativos ligam a camada  $F_1$  à camada  $F_2$ .

Os módulos  $ART_a$  e  $ART_b$  são ligados por uma rede de aprendizagem associativo (um mapa de unidades neurais). O treinamento da rede Fuzzy ARTMAP consiste da adaptação de sua arquitetura – número de neurônios do mapa e das camadas  $F_2^a$  e  $F_2^b$  – e da adaptação dos pesos. Quando a previsão feita pelo módulo  $ART_a$  é confirmada pelo módulo  $ART_b$ , isto é, quando os pesos do neurônio vencedor de cada camada comparados com o vetor de entrada ultrapassam o parâmetro de vigilância,

realiza-se a atualização dos pesos. Caso contrário, faz-se uma nova busca até encontrar um neurônio que satisfaça o *threshold* definido. Se nenhum neurônio satisfizer o parâmetro de vigilância, cria-se um novo neurônio na camada  $F_2$  do módulo que não encontrou o neurônio e, dependendo do módulo em que o neurônio foi criado, adiciona-se uma camada ou um neurônio por camada no mapa.

## 4 Trabalhos Relacionados

Em meados dos anos 80, apareceram os primeiros estudos sobre diagnósticos de falhas em sistemas de bombeio mecânico com emprego de conceitos de Inteligência Artificial (IA).

Em 1987, Foley e Svinos apresentaram o EXPROD, um programa de reconhecimento de padrão de cartas baseado em técnicas estatísticas e acoplado a um conjunto de regras definidas a partir da experiência em diagnósticos de sistemas de bombeio. Como amostra de dados, esse programa utiliza um grupo de *templates* de cartas dinamométricas para cada tipo de carta. Cada *template* é composto apenas por barras horizontais que representam o intervalo da posição esperada para uma dada carga.

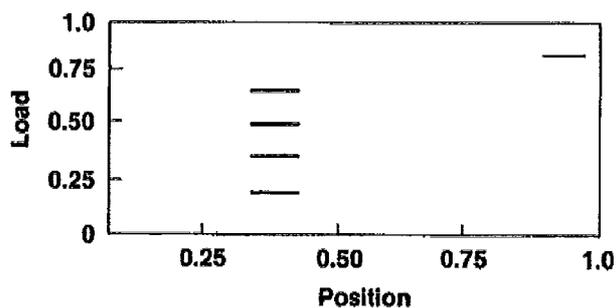


Figura 4.1. Template de carta com pancada de fluido do EXPROD (Foley e Svinos, 1987).

A figura 4.1 mostra um *template* de carta dinamométrica com pancada de fluido. Antes de ser comparada com os *templates*, a carta de entrada é normalizada para que seus valores de carga e posição também fiquem entre 0 a 1.

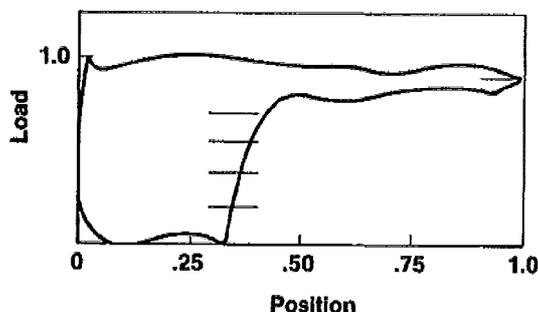


Figura 4.2. Comparação da carta com o template (Foley e Svinos, 1987).

Para se avaliar a similaridade entre uma carta e um *template*, verifica-se se as barras do *template* interceptam a borda da carta. A comparação de uma carta com o *template* de pancada de fluido pode ser vista na figura 4.2.

O grau de *matching*  $M$  entre uma carta e o *template* é dado pela fórmula:

$$M = \sum_{i=1}^n (k_i F_{wi}),$$
 onde  $k$  é igual a 1 (um) quando a barra interceptar a carta e, 0 (zero), caso contrário;  $F_w$  é um fator de ponderação (peso atribuído à barra que é definido por tentativas e erros); e  $n$  é o número de barras do *template*.

Ou seja, o grau de *matching* é a soma do peso das barras que interceptam a carta. O produto  $F_w K$  pode ser entendido como uma probabilidade, já que, quanto maior o número de barras interceptando a borda da carta dinamoétrica, maior a probabilidade da carta ser classificada no grupo do *template*.

O grau de *matching* pode ser aperfeiçoado aumentando-se a largura das barras horizontais (isto é, aumentando-se o intervalo da posição da carga) ou adicionando-se mais *templates*.

Outro algoritmo de reconhecimento de padrão de carta surgiu no fim dos anos 80. Conhecido como NUPRO (Keating, Laine e Jennings, 1989), o algoritmo utiliza uma técnica numérica de reconhecimento de padrões para classificar uma carta desconhecida a partir de um repositório de cartas conhecidas. A técnica consiste em selecionar  $n$  pontos (igualmente distribuídos ou não) de cada carta (das cartas conhecidas e da carta que se deseja classificar), normalizá-los e colocá-los em um vetor bidimensional. Em seguida, faz-se o produto escalar do vetor de cada carta conhecida com o vetor da carta desconhecida. Quanto maior o valor do produto escalar, maior o grau de similaridade entre as cartas. Assim, a carta desconhecida terá a classificação da carta do repositório cujo produto escalar entre seus vetores for o maior entre todos os realizados. Implementações do algoritmo NUPRO foram discutidas em (Keating, Laine e Jennings, 1991).

Em 1990, Dickinson e Jennings compararam alguns métodos de reconhecimentos de padrão para realizar a classificação de cartas dinamoétricas. O método *grade* (*grid*, em inglês), que descreve a forma da carta de fundo como uma matriz de zeros e uns, foi o método que apresentou os melhores resultados no diagnóstico de problemas em sistemas de bombeio mecânico. Nesse método, a carta de

entrada é normalizada e sobreposta em cima de uma grade com 200 células (20 x 10) com valor zero. Em seguida, todas as células que se encontram dentro do limite da carta recebem o valor 1 (um), como a carta da figura 4.3.

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0

Figura 4.3. Matriz de representação da carta de fundo (Dickinson e Jennings, 1990).

O próximo passo é comparar a matriz da carta de entrada com matrizes de cartas usadas como referências. Porém, as matrizes de referências são formadas de maneira diferente. Ao invés de receberem apenas zeros e uns, as matrizes de referência recebem valores graduais: valores maiores que 1 (um) dentro da carta e menores que 0 (zero) fora da carta, que aumentam ou diminuem, respectivamente, à medida que se afastam das bordas da carta, como é mostrado na figura 4.4.

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1
1	2	3	3	3	3	3	3	3	3	3	3	3	2	1	1	1	1	1	1	1
1	2	3	4	4	4	4	4	4	4	4	3	2	1	0	0	0	0	0	0	0
1	2	3	4	5	5	5	5	5	5	4	3	2	1	0	-1	-1	-1	-1	-1	-1
1	2	3	4	5	5	5	5	5	5	4	3	2	1	0	-1	-2	-2	-2	-2	-2
1	2	3	4	4	4	4	4	4	4	4	3	2	1	0	-1	-2	-3	-3	-3	-3
1	2	3	3	3	3	3	3	3	3	3	3	2	1	0	-1	-2	-3	-4	-4	-4
1	2	2	2	2	2	2	2	2	2	2	2	2	1	0	-1	-2	-3	-4	-5	-5
1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	-1	-2	-3	-4	-5	-5

Figura 4.4. Matriz de referência (Dickinson e Jennings, 1990).

O custo de *matching*  $C(A, B)$  entre uma matriz de entrada  $A$  (zeros e uns) e uma matriz de referência  $B$  (escala de cinza) é dado pela fórmula:

$$C(A, B) = \sum_{i=1}^{10} \sum_{j=1}^{20} |a_{ij} - f(b_{ij})|, \text{ onde } f(b_{ij}) = \begin{cases} 1, & \text{se } a_{ij} = 1 \text{ e } b_{ij} \geq 1 \\ 0, & \text{se } a_{ij} = 0 \text{ e } b_{ij} \leq 0 \\ b_{ij}, & \text{caso contrário.} \end{cases}$$

A carta de referência cuja matriz obtiver o menor custo de *matching* com a matriz da carta de entrada será a selecionada para representar a condição de fundo dessa carta.

Além do método *grade*, um outro método, também testado por Dickinson e Jennings (1990), destacou-se por apresentar bons resultados na identificação do diagnóstico a partir de cartas: o método do descritor de Fourier baseado na posição. O método consiste em representar a forma da carta de entrada e das cartas de referência por séries de Fourier e depois calcular a distância entre elas.

Em 1991, Schirmer, Gay e Toutain propuseram um sistema baseado em conhecimento composto por três diferentes módulos de análise: (1) sistema especialista, que usa reconhecimento de padrão estrutural para diagnosticar os problemas do sistema de bombeio mecânico; (2) módulo de comparação API, que compara a carta desconhecida com cartas dinamométricas contidas em um catálogo API<sup>8</sup>; e (3) um módulo de conclusões e recomendações.

O primeiro módulo utilizou o conceito de análise sintática para implementar o reconhecimento de padrão estrutural. Uma gramática foi definida para cada tipo de carta. A operação de reconhecimento consiste em se verificar se uma dada carta (ou seja, uma dada sentença) pode ser criada por uma determinada gramática.

No módulo de comparação API, *quadrees* – estruturas hierárquicas de dados usadas para armazenar e manipular imagens – são utilizados para comparar a carta de entrada com as cartas armazenadas no catálogo API de cartas de referência.

Para estruturar e sincronizar os módulos, foram utilizados os conceitos de *supervisor* e *blackboard* (quadro-negro). O supervisor – que pode ser interpretado como uma máquina de inferência do sistema – é responsável por disparar os módulos. Caso algum módulo identifique o problema do sistema de bombeio mecânico, ele escreverá a sua hipótese no quadro-negro. Após a execução do último módulo, inicia-se a fase de validação das hipóteses declaradas no quadro-negro, através da aplicação de metaregras pelo supervisor. Apenas as hipóteses confirmadas pelas metaregras tornar-se-ão diagnósticos. Os autores apresentam o seguinte exemplo: dado que um módulo identificou a hipótese “falha na ancoragem” para um poço que não possui coluna

---

<sup>8</sup> Catalog of Analog Computer Dynamometer Cards. API Bul 11L2, 1st Ed., American Petroleum Institute, Dallas Texas, December 1969

ancorada, impede-se que seja dado um diagnóstico incorreto pela definição da metaregra: *se, no quadro-negro, aparece a hipótese “falha na ancoragem” e se o poço não possui âncora, então a hipótese deve ser cancelada.*

No início dos anos 90, redes neurais com treinamento por retropropagação de erros começam a se destacar nos estudos de análise de cartas. Rogers *et al.* (1990) utilizam propriedades geométricas como entradas para uma rede neural. Além da área, perímetro e taxa de fineza da carta (todos normalizados), fazem uso também da distribuição da variância da transformada de Fourier que representa a carta. A rede neural – composta por 19 entradas, 279 neurônios na camada escondida e 2 neurônios na camada de saída – foi treinada com o algoritmo Gradiente Descendente durante 3.459 iterações. Como só havia 15 cartas dinamométricas disponíveis, que foram usadas no treinamento, não foi possível realizar testes com a rede.

Nazi *et al.* (1994) propuseram uma rede neural híbrida – que utiliza senóides e sigmóides como funções de ativação – também treinada com a técnica de retropropagação de erros. Essa rede é composta por 80 entradas, 60 neurônios na camada escondida e 11 neurônios na saída. O conjunto de entradas é composto por 60 pontos obtidos diretamente da carta e 20 pontos que representam a inclinação de alguns desses 60 pontos. A rede proposta é capaz de detectar 11 condições diferentes de problema ou qualquer combinação entre eles. Essa rede foi treinada, durante 20.000 iterações, com 176 cartas e testada com 20 cartas que não pertenciam ao conjunto de treinamento. Segundo o autor, a rede proposta classificou corretamente as 20 cartas de teste.

Em 1993, Martinez *et al.* apresentaram uma solução diferente baseada em redes neurais. Nesse trabalho, as cartas dinamométricas de fundo, calculadas a partir de cartas dinamométricas de superfície, são digitalizadas e normalizadas. De cada carta, são extraídos 100 pontos igualmente espaçados. É realizada uma compressão da informação em 30 pontos que apresentam as características mais relevantes da carta. Esses 30 pontos são, então, usados como entradas da rede neural. Trata-se de uma rede neural com função de ativação do tipo sigmóide, 8 neurônios na camada escondida e 21 neurônios na camada de saída. Após um treinamento de 10.000 ciclos (correspondendo a um total de quatro horas) com um conjunto de treinamento de 140 cartas, o sistema apresentou, sobre um conjunto de teste de 50 cartas, a precisão de 95% de acerto em seus diagnósticos. O sistema mencionado também oferece um módulo, baseado em

algoritmo genético, que fornece propostas para o projeto de instalação do sistema de bombeio mecânico, compatíveis com as restrições do equipamento e com a produção potencial do poço alimentada no sistema pelo usuário.

Rede neural competitiva auto-organizável também foi utilizada para classificação de cartas. Xu *et al.* (2007) compararam as redes competitivas com as redes supervisionadas, utilizando números diferentes de neurônios de entrada (54, 72, 108 e 216) e tempos diferentes de treinamento (0, 50, 100, 150, ..., 500). O modelo final escolhido possuía 216 neurônios de entrada, 5 neurônios na camada competitiva (que correspondem às condições do sistema de bombeio mecânico) e 100 ciclos de treinamento. De um total de 7887 cartas classificadas em 5 diferentes classes, foram selecionadas, aleatoriamente, 1550 cartas para treinamento da rede neural. As 6337 cartas restantes foram utilizadas para teste. As cartas não estavam distribuídas igualmente entre as classes, já que duas das cinco classes possuíam cerca de 83,85% da amostra de dados. Os autores declaram que a taxa média de acerto na classificação foi de 99,97% com desvio padrão de 0,0073.

A conclusão apresentada foi que o modelo de redes auto-organizáveis é mais estável, tem maior exatidão em diagnósticos e melhor capacidade de generalização, por não apresentar ponto mínimo local e por ter melhor taxa de convergência. Concluíram também que o aumento do tempo de treinamento e o aumento do número de neurônios de entrada melhoram a capacidade de classificação do modelo.

Nesse modelo, os neurônios da camada competitiva, como o nome diz, competem entre si para determinar o vencedor. No final da competição, somente o neurônio vencedor – aquele com maior entrada na rede – terá como saída o valor um, indicando a classificação da carta de entrada. Sendo  $p$  o vetor de entrada,  $w_j$  o vetor de pesos e  $b_j$  ( $b_j > 0$ ) o bias do neurônio  $j$ , o vetor de saída  $a$  da camada competitiva

é calculado como:  $a = \text{compet}(-\|w_j - p\| - b_j)$ , onde

$$\text{compet}(x) = \begin{cases} 1, & x = \max(x) \\ 0, & \text{caso contrário} \end{cases} .$$

Somente o vetor de pesos do neurônio vencedor é ajustado. Para isso, usa-se a taxa de aprendizado  $lr$  e a diferença entre o vetor normalizado de entrada  $p$  e o vetor de pesos  $w_j$  do neurônio vencedor  $j$  para ajustar os pesos da seguinte forma:

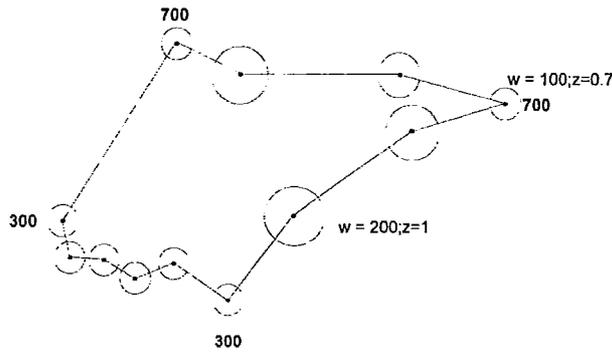
$\Delta w_j = lr.(p - w_j)$ . Faz-se, assim, que  $w_j$  vá em direção à  $p$ , diminuindo, cada vez mais, a diferença entre eles. No final, a matriz de pesos armazenará o vetor padrão de cada classe e o neurônio vencedor mostrará o resultado da classificação.

Em 1993, Alegre *et al.* apresentaram o SICAD, um sistema neural hierárquico para controle inteligente de sistemas de bombeio mecânico, composto por duas famílias de redes: Redes de Reconhecimento de Padrões (RNP) e Redes de Raciocínio Especialista (RNE). As RNP representam o nível inferior do sistema. Elas são responsáveis por reconhecer as novas cartas e ativar as redes RNE, que estão no nível superior.

O SICAD possui um controle local, realizado pelo Sistema Controlador de UB (SCUB), e um controle remoto, realizado pelo sistema conhecido como Escritório Central. O SCUB é responsável por obter a carta dinamométrica de superfície (CDS) através de sensores que fornecem a ele as cargas exercidas na coluna de hastes em cada posição do ciclo de bombeamento. Esse sistema calcula a carta dinamométrica de fundo (CDF) a partir da carta de superfície (CDS), compara-a com os padrões de carta nele armazenados e analisa as condições de bombeio. Esse sistema atua diretamente no controle da unidade de bombeio, sem intervenção humana, podendo interromper a operação do sistema, em caso de falha grave, ou reduzir a velocidade de bombeamento em caso de *pump-off*.

O sistema Escritório Central pode acessar o SCUB instalado no poço via rádio e realizar uma análise mais abrangente, utilizando também dados atuais e históricos do poço.

Para realizar o reconhecimento de padrões, pontos significativos (PS) normalizados de uma CDF desconhecida – pontos onde ocorrem importantes mudanças de direção da trajetória “Carga versus Ciclo” – são selecionados pelo SCUB e comparados com PS de cartas de referência armazenados em uma base de dados.



**Figura 4.5.** Padrão para CDF Pancada de Fluido definido pelo especialista (Corrêa, 1995).

Essas cartas de referência são construídas pelo especialista que, graficamente, define as coordenadas  $(x, y)$  de cada ponto significativo, o raio  $w$  do ponto em torno do qual a comparação ainda é aceitável e a relevância  $z$  desse ponto para o processo de decisão. Um exemplo de carta de referência definida por especialista pode ser visto na figura 4.5.

A confiança na hipótese  $c(s'_i \equiv s_i)$  de que um ponto significativo  $s'_i$  da CDF seja idêntico a um determinado ponto  $s_i$  do padrão pode ser calculada pela seguinte função nebulosa:  $c(s'_i \equiv s_i) = 1 - \frac{d_i}{2w}$ , onde  $d_i$  é a distância Euclidiana (cujo cálculo aparece na figura 4.6) entre o ponto  $s'_i$  e o ponto  $s_i$ ; e  $w$  é o raio do círculo medido do ponto  $s_i$  até o ponto onde o valor de confiança é igual a 0.5.

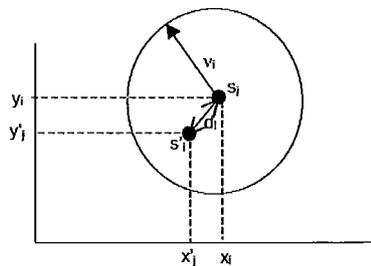


Figura 14 - Nodo

$$d_i = \sqrt{(x_i - x'_i)^2 + (y_i - y'_i)^2}$$

**Figura 4.6.** Cálculo da distância entre um ponto significativo  $s_i$  do padrão e um ponto significativo  $s'_i$  de uma carta ainda não classificada (Corrêa, 1998).

Compara-se, então, cada ponto  $s'_i$  da carta de entrada com todos os pontos  $s$  da carta de referência, adotando-se, para  $s'_i$ , o maior valor de confiança encontrado.

Para saber se uma carta de referência representa a classificação de uma nova carta, deve-se calcular a confiança na hipótese  $c(PS_i \equiv PS_j)$  de que o conjunto de pontos significativos  $PS_i$  da carta de entrada equivale ao conjunto de pontos significativos  $PS_j$  do padrão. A confiança é dada pela média obtida pelo somatório da confiança de cada ponto ponderada pela relevância  $z_i$  de cada um:

$$c(PS_i \equiv PS_j) = \frac{\sum_i c(s'_i \equiv s_i) z_i}{\sum_i z_i}$$

Dessa forma, quanto maior a relevância de um ponto, maior a sua contribuição para a confiança do padrão onde foi definido. Porém, sua relevância pode contribuir negativamente, ou seja, pode diminuir a confiança do padrão, caso a relevância seja um valor menor do que zero (Corrêa, 1998).

Todos os padrões (cartas de referências armazenadas) que apresentarem o valor de confiança acima do esperado são candidatos a representar a carta.

No SICAD, o conjunto de PS para cada carta de referência é definido por um especialista, pois, na época do desenvolvimento desse sistema, não havia uma base de dados com as cartas. Além disso, o custo computacional para treinar a rede com cartas de outros poços poderia ser muito alto. Nesse sistema, também fazem parte da responsabilidade do especialista, a definição da variabilidade de cada ponto e até a alteração do peso sináptico para refinar o conhecimento fornecido por ele inicialmente.

Corrêa (1995), em sua dissertação de mestrado, propõe, como ampliação da atuação do SICAD, a geração de propostas de ações para corrigir falhas e otimizar o sistema de bombeio mecânico, além do diagnóstico da carta, já antes realizado. Corrêa faz uso de agentes para realizar diferentes análises: verificação de existência atrito, presença de areia, profundidade da bomba, parafina e eficiência de bombeamento, entre outros. Há também agentes para confirmar as características relacionadas aos padrões de cartas (um exemplo pode ser visto na figura 4.7). Os agentes têm a forma de uma rede e utilizam a confiança supracitada (Alegre *et al.*, 1993) em uma de suas entradas.

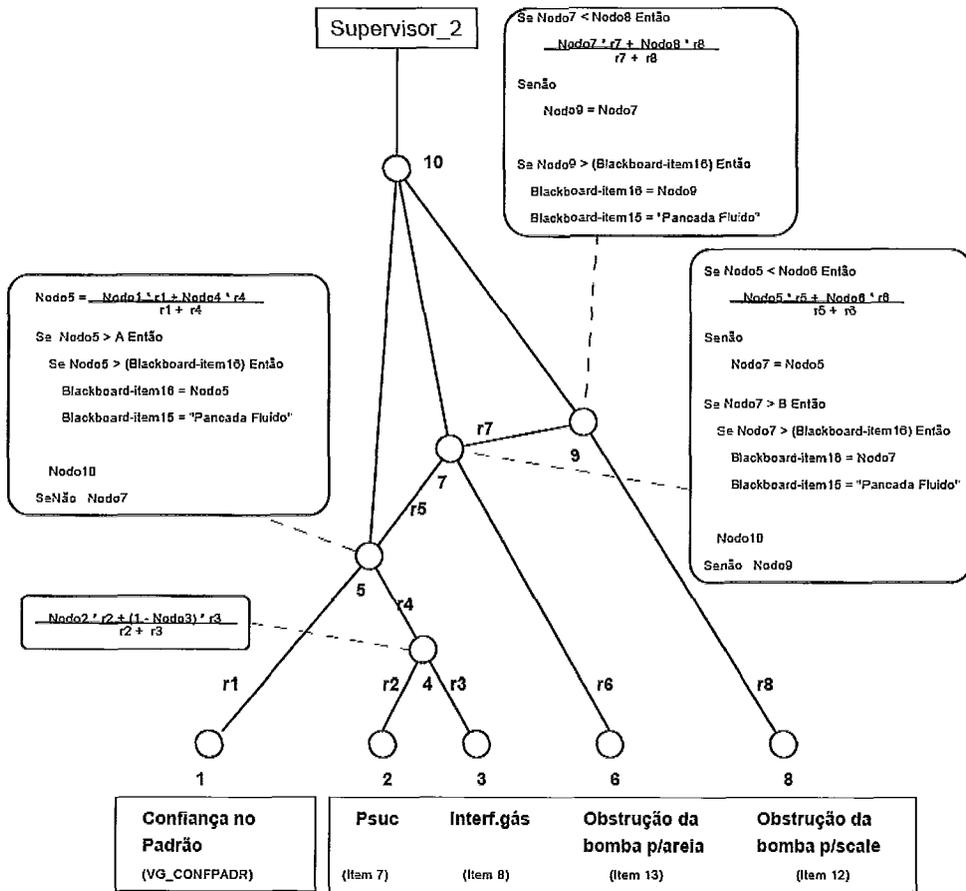


Figura 4.7. Agente que verifica Pancada de Fluido (Corrêa, 2003).

Os agentes são disparados por supervisores – estruturas projetadas para ativar agentes na ordem programa pelo especialista – e comunicam-se entre si utilizando o conceito de *blackboard* (Cerqueira *et al.*, 2002).

O aumento do tempo médio entre falhas<sup>9</sup> foi um dos principais resultados desse trabalho – apresentados em (Corrêa, 2003) –, por ser um ganho que pode ser mensurado.

Para obter resultados ainda melhores, um novo método para a geração de padrões de carta dinâmométrica de fundo foi criado: ao invés de se utilizar a tabela de padrões definida manualmente pelo especialista – explicado em (Corrêa, 1998) –, novos padrões foram criados a partir da extração dos pontos relevantes de cartas previamente classificadas (Schnitman *et al.*, 2003). Outra proposta para geração dos padrões de carta de fundo (CDF), baseada em algoritmos genéticos, foi apresentada em (Schnitman *et al.*, 2004).

<sup>9</sup> Tempo Médio Entre Falhas - também conhecido pelo termo inglês *Mean Time Between Failures (MTBF)* - é um índice que representa a confiabilidade de um dispositivo ou aparelho.

## 5 Metodologia para Definição da Rede Neural

Esta seção apresenta a metodologia utilizada para definição, treinamento e simulação da rede neural.

### 5.1 Pré-processamento dos dados

Uma carta dinamométrica de fundo (CDF) é composta, em geral, por 100 pontos. Cada ponto representa um par de coordenadas carga versus posição. Nesta seção, será descrito o pré-processamento de dados, que inclui a normalização dos pontos da carta e a redução dos atributos de entrada da rede neural.

#### 5.1.1 Normalização dos dados

Conforme explicado no Capítulo 3, as variáveis de entrada de uma rede neural devem ser normalizadas para evitar a paralisia da rede.

Para que os valores de carga e posição assumissem valores entre 0 e 1, a normalização adotada foi a normalização MIN-MAX, na qual se subtrai o valor mínimo do valor a ser normalizado e, em seguida, divide-se o resultado pela amplitude dos valores ( $valor_{máximo} - valor_{mínimo}$ ), da seguinte forma:

$$valor_{normalizado} = \frac{valor_{não-normalizado} - valor_{mínimo}}{valor_{máximo} - valor_{mínimo}}$$

Durante a identificação de cartas dinamométricas de fundo, é importante que imagens com diferentes tamanhos, mas com o mesmo formato, sejam reconhecidas como pertencentes à mesma classe. Por isso, a normalização adotada neste trabalho será realizada por instância e não por atributos, como normalmente ocorre. Mais especificamente, a normalização será realizada para todas as cargas de uma CDF em relação às cargas máxima e mínima registradas. Da mesma forma, as posições da CDF serão normalizadas em relação à posição máxima e à posição mínima encontradas naquela carta.

Os passos para normalização dos dados de uma CDF podem ser resumidos no seguinte algoritmo:

Para cada  $CDF_{\text{n\~{a}o-normalizada}}$ , faa:

1. Encontre a maior e a menor carga ( $C_{\text{m\~{a}xima}}$  e  $C_{\text{m\~{i}nima}}$ ) registrada na  $CDF_{\text{n\~{a}o-normalizada}}$ .
2. Encontre a maior e a menor posio ( $P_{\text{m\~{a}xima}}$  e  $P_{\text{m\~{i}nima}}$ ) registrada na  $CDF_{\text{n\~{a}o-normalizada}}$ .
3. Para cada ponto ( $C_{\text{n\~{a}o-normalizada}}$ ,  $P_{\text{n\~{a}o-normalizada}}$ ) da  $CDF_{\text{n\~{a}o-normalizada}}$ , faa:

a. Calcule a carga normalizada: 
$$C_{\text{normalizada}} = \frac{C_{\text{n\~{a}o-normalizada}} - C_{\text{m\~{i}nima}}}{C_{\text{m\~{a}xima}} - C_{\text{m\~{i}nima}}}$$

b. Calcule a posio normalizada: 
$$P_{\text{normalizada}} = \frac{P_{\text{n\~{a}o-normalizada}} - P_{\text{m\~{i}nima}}}{P_{\text{m\~{a}xima}} - P_{\text{m\~{i}nima}}}$$

- c. Crie o ponto normalizado ( $C_{\text{normalizada}}$ ,  $P_{\text{normalizada}}$ ) e adicione-o na  $CDF_{\text{normalizada}}$ .

## 5.1.2 Reduo de atributos

O tamanho de uma rede neural aumenta à medida que se adicionam variáveis de entrada a ela, pois aumenta-se o número de conexões – número de pesos – entre a camada de entrada e a camada escondida de neurônios. Como já visto, o crescimento do número de pesos de uma rede aumenta do risco de *overfitting* e torna necessário um conjunto de treinamento maior.

Tudo isso faz com que o tempo de treinamento da rede também aumente. A esses problemas, Berry e Linoff (2004) acrescentam o fato de que o excesso de entradas deixa os pesos menos próximos de seus valores ótimos. Com todas essas desvantagens em se ter um grande número de entradas, não é difícil deduzir que o uso das 200 coordenadas como entradas da rede não seria uma boa solução.

A primeira redução de atributos realizada sobre as 200 coordenadas de carga e posição foi baseada, inicialmente, na observação dessas coordenadas de uma forma diferente da usual.

Normalmente, um especialista em bombeio mecânico visualiza o formato da carta dinamométrica de fundo (CDF) – gráfico carga x posição como o que aparece na figura 5.1 – para poder identificar a condição de bombeio naquele ciclo.

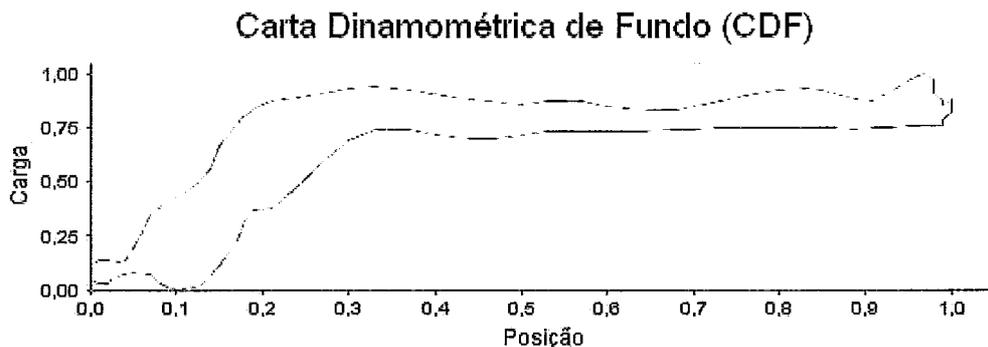


Figura 5.1. Exemplo de uma CDF normalizada: carga x posição.

A CDF nada mais é do que um gráfico que apresenta duas variáveis – carga e posição – que variam no decorrer de um ciclo de bombeio mecânico, isto é, variam no intervalo de tempo de um ciclo. Assim, para se identificar uma condição de bombeio, ao invés de analisar o formato da carta, poder-se-ia observar dois outros gráficos: o gráfico da variação da carga durante um ciclo de bombeio (Figura 5.2a) e o gráfico do deslocamento da coluna de hastes durante o mesmo intervalo de tempo (Figura 5.2b).

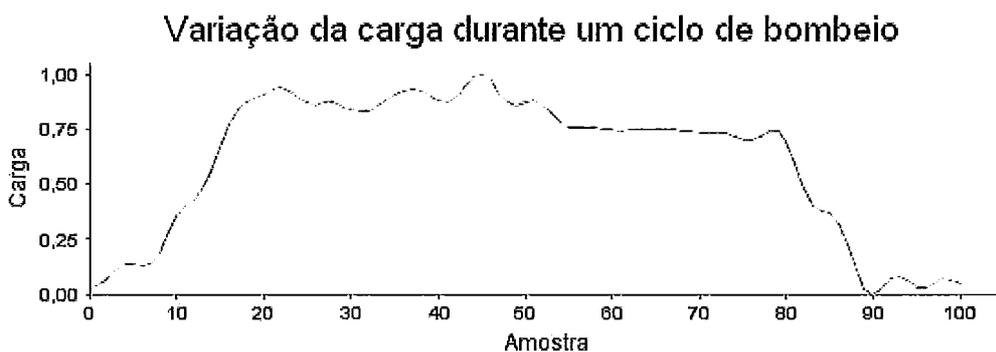
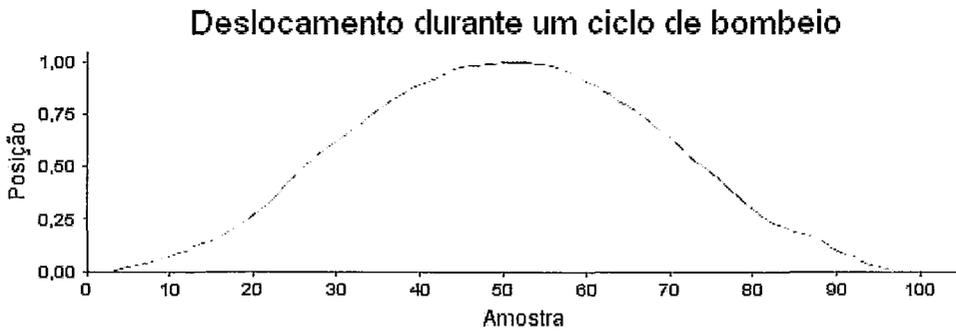
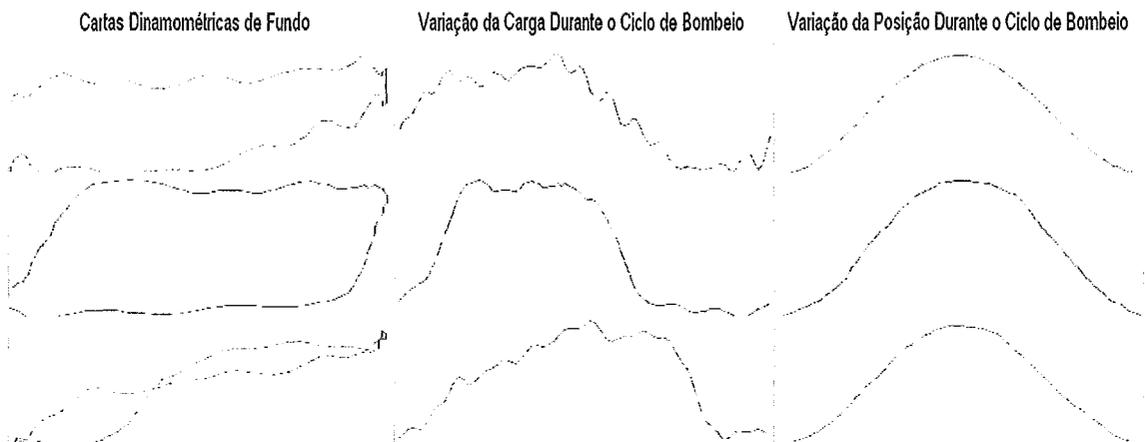


Figura 5.2a. Cargas da CDF da Figura 1: carga x amostra no tempo.



**Figura 5.2b.** Posições da CDF da Figura 1: posição x amostra no tempo.

Em geral, havendo ou não problemas no bombeio, a curva formada pelo deslocamento durante um ciclo de bombeio (figura 5.2b) tem o formato semelhante ao de uma gaussiana. Essa curva pode sofrer algumas alterações, principalmente devido a falhas na medição durante um ciclo. Apesar disso, a curva que tem o comportamento menos regular em relação a diferentes condições de bombeio é aquela que descreve a variação da carga durante um ciclo de bombeio, como pode ser visto na Figura 5.3.



**Figura 5.3.** Exemplos de cartas e suas respectivas cargas e posições durante o ciclo de bombeio.

A Figura 5.3 mostra exemplos de cartas dinamométricas de fundo (CDF), cada uma ao lado dos respectivos gráficos de variação da carga e de variação da posição que deram origem àquela carta. Embora cada CDF represente uma condição de bombeio diferente, todas apresentam curvas semelhantes de deslocamento durante o ciclo de bombeio. A curva que mostra a variação da carga, no entanto, varia bastante de uma CDF para a outra.

O comportamento regular do deslocamento em diferentes situações nos leva a concluir que, embora seja mais fácil para um especialista identificar a condição de bombeio pela carta dinamométrica, para uma rede neural, pode ser mais fácil reconhecê-la apenas pelas cargas da CDF.

Segundo Carvalho (2001), a redução do número de variáveis pode ser feita baseada no sentimento do analista ou em técnicas estatísticas. Até o momento, tentou-se explicar o “sentimento” de que pode ser possível identificar uma falha no bombeio mecânico apenas a partir das coordenadas de carga de uma CDF. Para que essa intuição seja comprovada não apenas pelos poucos exemplos apresentados na figura 5.3, realizou-se uma análise estatística a partir das coordenadas de uma amostra de 400 cartas.

Para a realização dessa análise, foram calculados a média e o desvio padrão para cada um dos 200 atributos normalizados, como mostra a tabela 5.1.

Cartas	Carga 1	Posição 1	Carga 2	Posição 2	...	Carga 100	Posição 100
CDF 1	0,20	0,00	0,24	0,01	...	0,15	0,02
CDF 2	0,07	0,00	0,08	0,00	...	0,06	0,01
...	...	...	...	...	...	...	...
CDF 400	0,64	0,00	0,72	0,00	...	0,58	0,01
<b>Média</b>	0,1511	0,0100	0,1725	0,0066	...	0,1337	0,0158
<b>Desvio Padrão</b>	0,1532	0,0170	0,1655	0,0145	...	0,1425	0,0199

Tabela 5.1. Cálculo da média e desvio padrão de cada um dos 200 atributos.

Para comparar o comportamento das cargas em relação ao das posições, os seguintes cálculos foram efetuados:

$$MédiaGeral_{Todos} = \frac{Média_{Carga1} + Média_{Posição1} + \dots + Média_{Carga100} + Média_{Posição100}}{200}$$

$$MédiaGeral_{Carga\ a} = \frac{Média_{Carga1} + Média_{Carga2} + \dots + Média_{Carga100}}{100}$$

$$MédiaGeral_{Posição} = \frac{Média_{Posição1} + Média_{Posição2} + \dots + Média_{Posição100}}{100}$$

$$DesvPadMédio_{Todos} = \frac{DesvPad_{Carga1} + DesvPad_{Posição1} + \dots + DesvPad_{Carga100} + DesvPad_{Posição100}}{200}$$

$$DesvPadMédio_{Carga\ a} = \frac{DesvPad_{Carga1} + DesvPad_{Carga2} + \dots + DesvPad_{Carga100}}{100}$$

$$DesvPadMédio_{Posição} = \frac{DesvPad_{Posição1} + DesvPad_{Posição2} + \dots + DesvPad_{Posição100}}{100}$$

Os resultados estão expostos na tabela 5.2. Os valores que são importantes para a análise são as médias dos desvios padrão encontrados. O desvio padrão médio das posições, como esperado, é inferior ao desvio padrão médio de todos os atributos (menos da metade) e inferior ao desvio padrão médio das cargas (menos de um terço), o que mostra que as posições estão bem mais concentradas próximas à média do que as cargas. Confirmada a hipótese de que as cargas possuem maior desvio padrão médio, tem-se mais um indício de que elas podem ser suficientes para a identificação da condição operacional do bombeio mecânico.

	<b>Todos</b>	<b>Posições</b>	<b>Cargas</b>
<b>Média Geral</b>	0,5153	0,4939	0,5368
<b>Desvio Padrão Médio</b>	0,1225	0,0591	0,1859

**Tabela 5.2.** Média geral e desvio padrão médio dos atributos agrupados.

Para mostrar que, eliminando-se os atributos de posição, o erro médio quadrático (MSE) mensurado na simulação da rede não sofrerá uma alteração significativa e que o tempo de treinamento será reduzido extremamente, foram realizados testes com 3 redes: uma rede com os 200 atributos como entradas (100 cargas e 100 posições); outra com apenas as 100 posições; e, finalmente, uma rede com somente as 100 cargas como entradas.

	<b>Todos</b>	<b>Posições</b>	<b>Cargas</b>
<b>MSE</b>	0,0590	0,0715	0,0684
<b>Tempo (s)</b>	708.878,40	95.724,73	81.486,36

**Tabela 5.3.** MSE e Tempo de Treinamento das redes com diferentes entradas.

A tabela 5.3 mostra o erro médio e o tempo total de treinamento. No apêndice A, encontram-se os erros medidos em detalhes. Pode-se ver que o erro não se altera de forma significativa com a redução de atributos. O tempo de treinamento, porém, sofre uma queda expressiva. A comparação entre os tempos de treinamento pode ser melhor visualizada no gráfico 5.1.

Por tudo o que foi apresentado, conclui-se que a eliminação dos atributos de posição da carta fornecerá ganho de performance à rede neural.

### Tempo de Treinamento por Quantidade e Tipo de Atributos

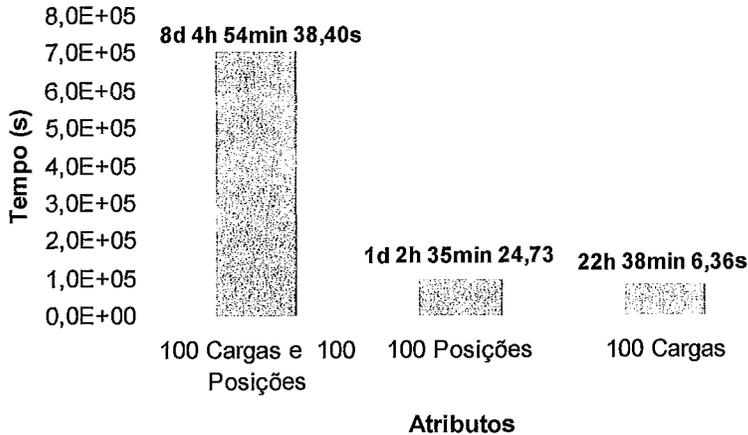


Gráfico 5.1. Tempo de treinamento por quantidade e tipos de atributos.

## 5.2 Definição da Rede Neural

Para se criar uma rede neural, é necessário especificar a sua topologia:

- Número de camadas escondidas;
- Número de neurônios em cada camada escondida;
- Número de saídas da rede;
- Função de ativação utilizada nos neurônios das camadas escondidas e da camada de saída.

A seguir, será apresentado como cada item acima foi definido.

### 5.2.1 Número de camadas escondidas

Dispensados os atributos de posição, o problema de identificar a condição de bombeio a partir da carta tornou-se um problema de aproximação de função, isto é, deseja-se encontrar a função que modela a variação das cargas durante um ciclo de bombeio para cada situação específica.

Sabendo-se que a variação da carga em um ciclo representa uma função contínua e visto, no capítulo de Redes Neurais, que redes MPL com apenas uma camada

intermediária (escondida) são aproximadores universais para funções contínuas, somente redes com uma camada intermediária foram utilizadas neste trabalho.

### **5.2.2 Número de neurônios na camada escondida**

Para se determinar a quantidade ideal de neurônios na camada escondida, serão realizados testes com diferentes números de neurônios na camada intermediária e a rede escolhida será aquela com a quantidade de neurônios que apresentar o menor erro médio quadrático na classificação das cartas.

### **5.2.3 Número de saídas da rede**

Os dados serão distribuídos em quatro classes: Normal, Pancada de Fluido, Interferência de Gás e Outra Classificação. Portanto, a rede neural terá quatro saídas, cada uma representando uma classe, isto é, indicando se a carta pertence ou não àquela classe. Para o treinamento da rede, adotou-se que cada saída terá valor 1 (um) quando a carta pertencer àquela classe e, valor 0 (zero), caso contrário. Assumiu-se também que cada carta só poderá pertencer a uma classe.

Outra abordagem que será avaliada neste trabalho é a utilização de um conjunto de redes neurais, na qual cada rede será especializada em uma condição específica de bombeio. Neste caso, cada rede terá apenas uma saída. Uma rede especializada em reconhecer Pancada de Fluido, por exemplo, deverá fornecer uma saída com valor próximo a 1 (um) quando a carta representar uma pancada e próximo a 0 (zero) quando representar outra situação operacional.

### **5.2.4 Função de ativação**

A função de ativação adotada nas camadas intermediárias será a função tangente hiperbólica, visto que, no aprendizado da rede, esse tipo de função tende a convergir mais rápido do que uma função do tipo logística.

Na camada de saída, a função adotada será a função linear para que a resposta da rede neural seja a combinação linear da saída dos neurônios da camada escondida.

## 5.3 Treinamento da Rede Neural

Para se treinar uma rede neural, diversos parâmetros devem ser definidos. Dentre eles, estão:

- Número de ciclos (épocas) de treinamento;
- Conjunto de treinamento;
- Algoritmo de treinamento;
- Parâmetros específicos do algoritmo de treinamento escolhido, como, por exemplo, a *taxa de aprendizado* e o *momento*, no caso do algoritmo do Gradiente Descendente.

A seguir, será explicado como esses parâmetros foram definidos.

### 5.3.1 Número de ciclos de treinamento

Para se determinar o número adequado de ciclos (épocas) de treinamento de forma a obter uma boa generalização, adotou-se o método de Parada Antecipada (*Early Stopping*). Para isso, foi necessário que a amostra de dados fosse dividida em três conjuntos:

- Conjunto de Treinamento (Estimação) – padrões usados para modificar os pesos durante o treinamento. Neste trabalho, foram destinados 70% dos padrões a esse conjunto;
- Conjunto de Validação – padrões usados para verificar o problema de *overfitting* e, conseqüentemente, para definir o número de ciclos ideal. Foram reservados 20% dos padrões para esse conjunto;
- Conjunto de Teste – padrões para testar o desempenho do modelo final – 10% dos padrões fizeram parte do conjunto de teste.

### 5.3.2 Conjunto de treinamento

Para garantir que o conjunto de treinamento escolhido fosse o mais adequado para fornecer o melhor desempenho na generalização, realizou-se a validação cruzada dos conjuntos. Os 400 padrões foram separados em 10 conjuntos (I, II, III,..., X) de 40

padrões, escolhendo-se um conjunto para teste, dois conjuntos para validação e sete para treinamento. Foram testadas as 6 combinações a seguir:

Rede	Treinamento	Validação	Teste
1	I a VII	VIII e IX	X
2	I a VII	IX e X	VIII
3	IV a X	II e III	I
4	IV a X	I e II	III
5	II a VIII	IX e X	I
6	III a IX	I e II	X

Em todos os testes, cada uma das seis redes acima foi executada cinco vezes – totalizando 30 execuções – para diferentes quantidades de neurônio na camada escondida (de 1 a 50 neurônios). A explicação resumida da execução dos testes pode ser vista através do algoritmo:

NUM\_REDES = 6;

EXECUCOES\_POR\_REDE = 5;

NUM\_MAX\_NEURONIOS = 50;

para cada número de neurônios  $n$ , de 1 até NUM\_MAX\_NEURONIOS, faça:

para cada execução  $e$ , de 1 até EXECUCOES\_POR\_REDE, faça:

para cada rede  $r$ , de 1 até NUM\_REDES, faça:

Cria a rede  $r$  com  $n$  neurônios na camada escondida;

Treina a rede  $r$ ;

Salva performance da rede  $r$ ;

fim-para;

Calcula performance média para a execução  $e$ ;

fim-para;

Calcula performance média para uma rede com  $n$  neurônios na camada escondida;

fim-para.

### 5.3.3 Algoritmo de treinamento

O algoritmo de treinamento adotado foi o de Levenberg-Marquardt. Esse algoritmo, em geral, converge mais rápido – utiliza um menor número de ciclos – que os demais quando se trata de problemas de aproximação de função e quando a rede utilizada não é grande (poucas centenas de pesos) (Demuth, Beale e Hagan, 2008).

Para verificar se o desempenho da rede seria alterado caso o algoritmo de treinamento fosse substituído por outro, alguns testes foram realizados com os dois algoritmos de treinamento mais comuns: o Levenberg-Marquardt (LM) e o método do Gradiente Descendente (GD). Nesses testes, percebeu-se que o tempo de um ciclo do LM é mais demorado que o do GD, já que o LM despende um tempo bem maior para o treinamento, mesmo convergindo em um número inferior de ciclos. Utilizando uma configuração semelhante ao do GD, em geral, o LM converge para erros menores. Os resultados dos testes com os dois algoritmos são apresentados, em detalhes, no Apêndice A.

## 5.4 Avaliação dos Resultados

A métrica MSE (*Mean Squared Error*), apresentada no Capítulo 3, será aplicada para comparar o desempenho entre as redes supervisionadas. Para a avaliação final da rede que apresentar o menor MSE, é necessário definir um valor limite (*threshold*) para se discretizar as saídas da rede, como será explicado a seguir.

### 5.4.1 Discretização da saída

Uma rede neural do tipo MLP, em geral, não fornece valores binários em suas saídas, exceto se for utilizada uma função do tipo Degrau como função de ativação. No caso de redes usadas para classificação, como a rede que se deseja obter com este trabalho, é necessário transformar suas saídas contínuas (*fuzzy*) em saídas discretas (*crisp*). Desse modo, precisa-se definir, para cada saída, um valor limite (*threshold*) a partir do qual o padrão será considerado pertencente à classe representada por aquela saída.

O desempenho da rede neural varia em função do *threshold*. Se o valor do *threshold* for equivalente ao maior valor fornecido pela rede, nenhum padrão será

classificado como pertencente à classe. No caso oposto, se o *threshold* for equivalente ao menor valor fornecido pela rede, todos os padrões serão classificados como pertencentes àquela classe. O objetivo é, portanto, encontrar o valor de *threshold* que maximize o número de padrões classificados corretamente.

O presente trabalho adotou a seguinte abordagem para definição do *threshold*: diferentes valores de *threshold* foram gerados e, para cada um, verificou-se a taxa de acerto obtida na classificação das cartas do conjunto de teste. Escolheu-se, então, o valor que forneceu a maior taxa de acerto.

O *threshold* que retorna o melhor desempenho da rede pode ser definido a partir de outras técnicas mais sofisticadas. Entre elas, pode-se citar a aplicação de algoritmos genéticos e a análise feita a partir de Curvas ROC<sup>10</sup> (*Receiver Operating Characteristic*).

## 5.4.2 Aperfeiçoamento dos resultados

Após se obter o resultado inicial com a rede configurada e treinada conforme os passos descritos neste capítulo, algumas técnicas serão adotadas para o aprimoramento dos resultados. A primeira delas será a continuação da redução de atributos realizadas no pré-processamento, pois acredita-se que a utilização da quantidade de 100 variáveis como entrada da rede ainda é um número grande o suficiente para levar ao supertreinamento (*overfitting*) da rede, ainda mais quando se tem uma amostra de dados pequena para treinamento.

Serão analisadas também redes do tipo SIM-NÃO, isto é, as redes especializadas em determinadas situações operacionais. Por fim, serão realizados testes com redes Fuzzy ARTMAP e com redes não supervisionadas, como os mapas de Kohonen, para verificar se resultam em erros menores de classificação de carta do que os fornecidos pelas tradicionais MLP.

---

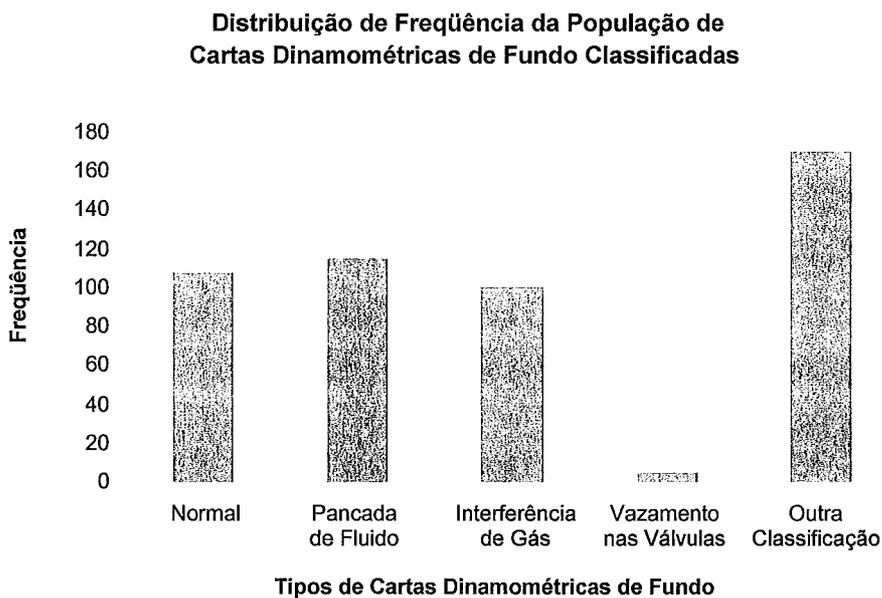
<sup>10</sup> A curva ROC é um gráfico que apresenta a dependência entre a sensibilidade – indicada pela fração de verdadeiro positivo (FVP) – e a especificidade – fração de verdadeiro negativo (FVN) – de um classificador (Sovierzoski, Argoud e Azevedo, 2008). Uma curva ROC é uma descrição empírica da capacidade do sistema de diagnóstico poder discriminar entre dois estados em um universo, onde cada ponto da curva, determinado por um diferente valor de corte (*threshold*), representa uma razão distinta entre a FVP e a FFP. (Metz, 1986, citado por Braga, 2000).

## 6 Validação da Rede Neural

Esta seção apresenta a seleção da amostra de dados e o resultado dos testes realizados, seguindo a metodologia mencionada no capítulo anterior.

### 6.1 Definição da amostra de dados utilizada

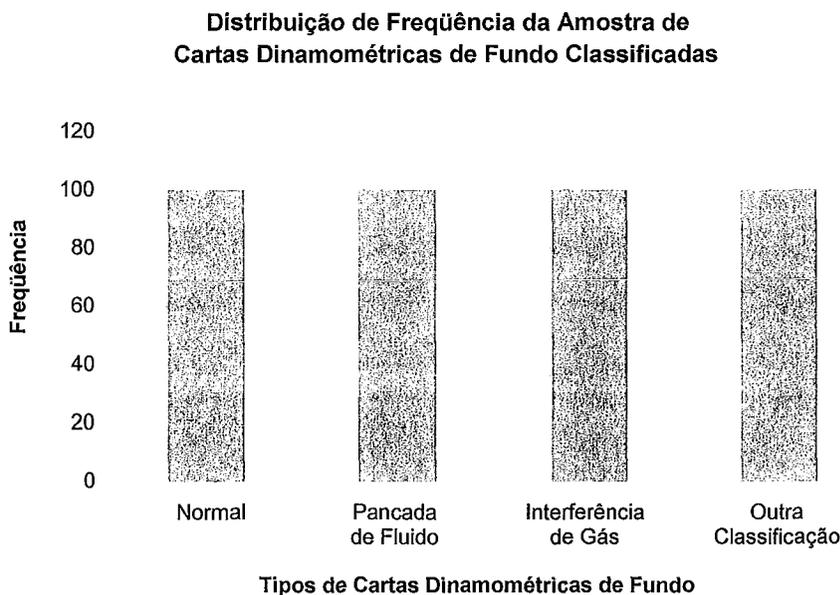
O acervo disponível de cartas classificadas é restrito: apenas 500 cartas. Essas cartas estão classificadas como: Normal, Pancada de Fluido, Interferência de Gás, Vazamento nas Válvulas e Outra Classificação. A distribuição de frequência das classificações da população de cartas dinamométricas de fundo (CDF) é apresentada no gráfico 6.1.



**Gráfico 6.1.** Distribuição de frequência da população de CDF.

Se uma rede neural for treinada com classes cujas frequências sejam muito diferentes uma das outras, como ocorre na população mostrada acima, ela tenderá a se especializar nas classes que apresentarem as maiores frequências. Dessa forma, se treinarmos uma rede utilizando toda a população de cartas, a rede acertará mais ao afirmar que uma carta representa ou não uma situação Normal do que ao afirmar que uma carta indica ou não Vazamento nas Válvulas.

Para realizar o treinamento da rede neural sem que ela ficasse especializada em determinadas classes, retirou-se, da população de cartas, uma amostra que contém a mesma quantidade de padrões de cartas para cada classe, como é indicado pela distribuição de freqüência do gráfico 6.2.



**Gráfico 6.2.** Distribuição de freqüência da amostra de CDF.

Como a freqüência de cartas que indicavam Vazamento nas Válvulas era insuficiente para que a rede aprendesse a reconhecer tal tipo de problema, essas cartas foram incorporadas à classe Outra Classificação. Em seguida, para se obter uma amostra com o mesmo número de ocorrências para cada tipo de carta, optou-se por selecionar 100 cartas de cada classe, visto que todas as classes apresentavam freqüência maior que essa quantia. A amostra utilizada neste trabalho contém, portanto, 400 exemplares de cartas.

## 6.2 Resultado Inicial

Inicialmente, treinou-se a rede neural com as 100 cargas como variáveis de entrada. O MSE obtido com o conjunto de teste foi de 0,0684, com um desvio padrão de 0,0212. Como se pode ver no gráfico 6.3, o menor erro médio (0,0532) foi encontrado com 25 neurônios na camada escondida. Os erros de treinamento, validação e teste podem ser vistos com mais detalhes no Apêndice A.

### MSE em Função do Tamanho da Camada Escondida

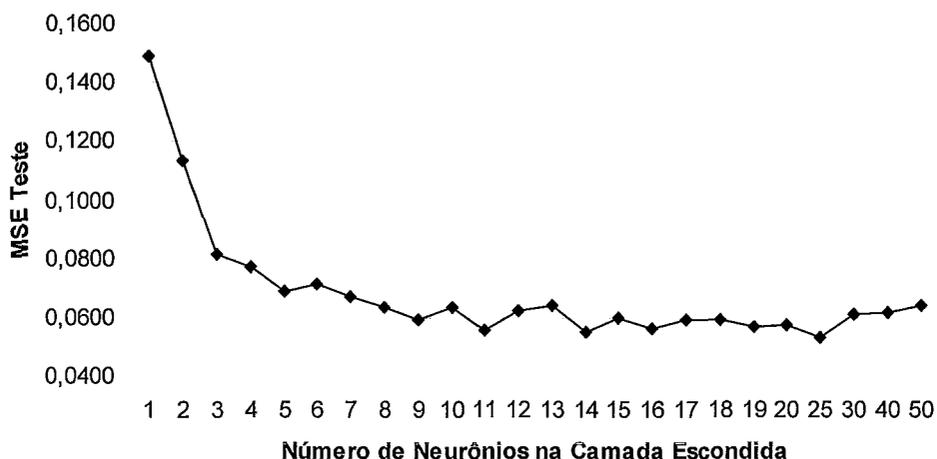


Gráfico 6.3. MSE em função do tamanho da camada escondida.

Embora a rede que utiliza as 100 cargas como variáveis de entrada tenha apresentado um índice de acerto de quase 94%, ainda era necessário reduzir o número de atributos, dado o tamanho limitado do conjunto de treinamento. A seguir, são apresentados os métodos para seleção dos atributos e o resultado das reduções.

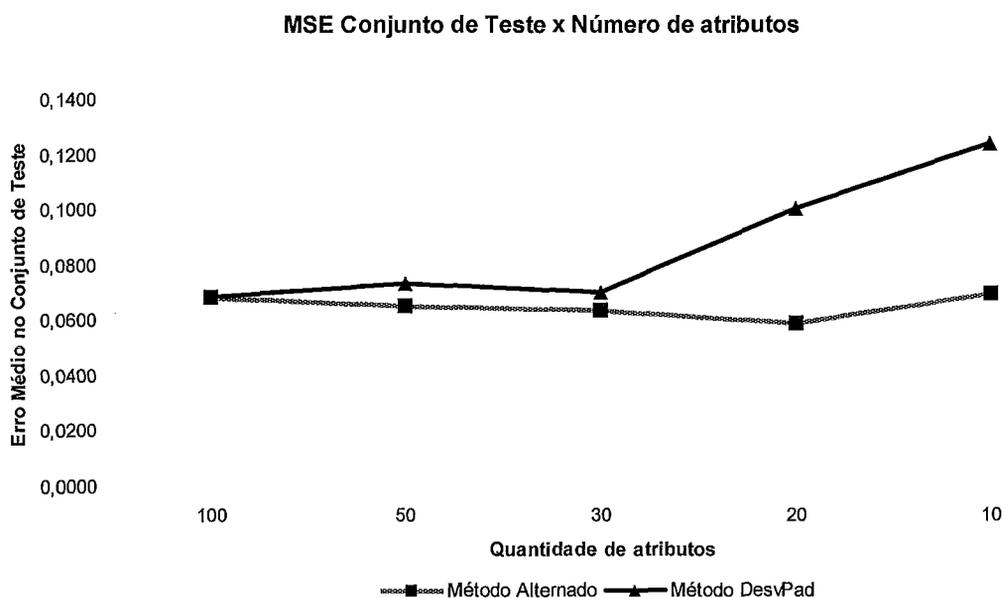
## 6.3 Redução de Atributos

Para reduzir o número de variáveis de entrada da rede, foram propostos dois métodos de seleção de atributos:

- Método Alternado – Os atributos são escolhidos alternadamente. Por exemplo, para escolher 10 atributos, selecionou-se 1 a cada 10 atributos; para escolher 20 atributos, selecionou-se 1 a cada 5 atributos; e, assim, por diante.
- Método DesvPad – Todos os atributos são ordenados segundo o seu desvio-padrão e, em seguida, são escolhidos aqueles que têm maiores desvios. Por exemplo, para escolher 10 atributos, selecionou-se os 10 atributos com maior desvio; para escolher 20, selecionou-se os 20 atributos com maior desvio; e, assim, por diante.

A partir do conjunto inicial de 100 cargas, foram realizadas 4 reduções – 50 cargas, 30 cargas, 20 cargas e 10 cargas – pelos métodos Alternado e DesvPad. O

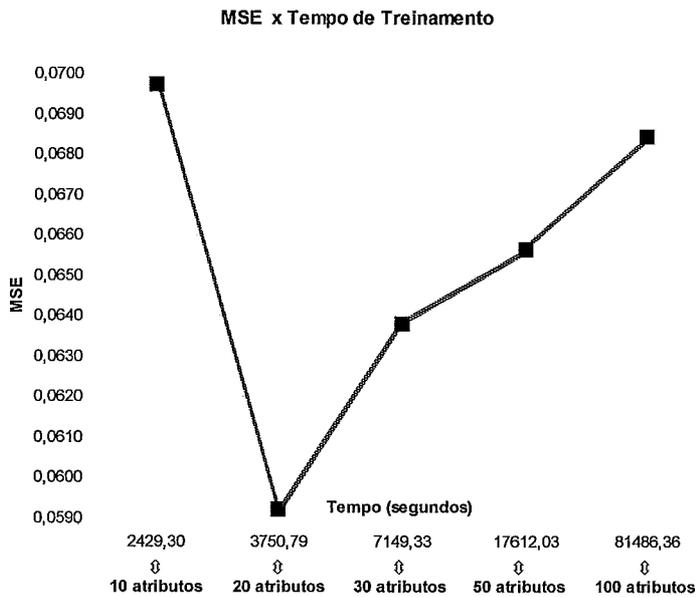
gráfico 6.4 mostra o resultado das reduções obtidas pela aplicação de cada método. O MSE mensurado na simulação da rede com 100 entradas é apresentado no gráfico para verificar o ganho obtido com a redução de 100 para 50 atributos.



**Gráfico 6.4.** Erro médio de validação em função do número de atributos.

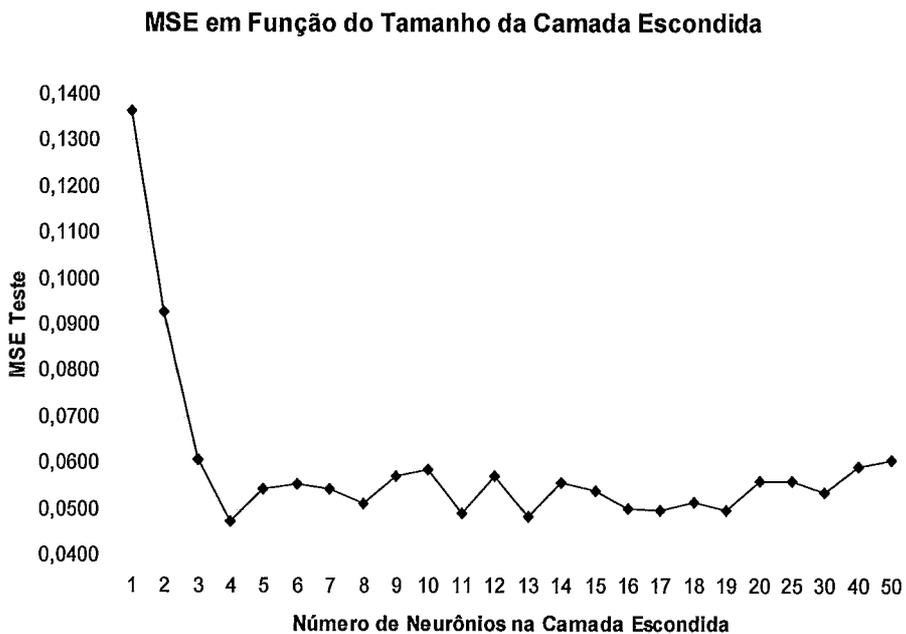
Pelo gráfico, pode-se ver que a rede reduzida pelo algoritmo DesvPad não teve bons resultados, pois, à medida que os atributos foram sendo eliminados, o erro obtido com a eliminação pelo método DesvPad aumentou consideravelmente. Já a redução de atributos pelo método Alternado apresentou bons resultados até alcançar 20 atributos. Logo, optou-se por utilizar uma rede neural com 20 variáveis de entrada definidas pelo método Alternado de seleção de atributos.

O gráfico 6.5 mostra o erro médio quadrático em função do tempo de treinamento, que aumenta conforme se aumenta o número de entradas. Tal dado permite verificar que a escolha pela rede de 20 entradas também é uma boa opção em relação ao tempo de treinamento.



**Gráfico 6.5.** MSE em função do tempo de treinamento.

O gráfico 6.6 mostra o comportamento da rede neural, que utiliza os 20 atributos selecionados pelo método Alternado como entradas, para diferentes números de neurônios na camada escondida. O menor erro médio (0,0474) foi encontrado com 4 neurônios na camada escondida. O menor MSE encontrado em todas as execuções foi de 0,0061 com 4 neurônios na camada escondida e utilizando a Rede 1.



**Gráfico 6.6.** MSE em função do tamanho da camada escondida

## 6.4 Discretização das Saídas

Conforme apresentado, a rede neural com melhor desempenho apresentou o MSE de 0,0061. É necessário, no entanto, discretizar as saídas da rede para se obter o percentual de acerto na classificação das cartas do conjunto de teste. Para isso, é preciso definir um valor limite (*threshold*), a partir do qual uma carta será considerada pertencente à classe que aquela saída representa.

A curva do gráfico 6.7 foi gerada para apresentar o comportamento da taxa de acerto em função de diferentes valores de *threshold*. Assumiu-se, para simplificar o problema, que a mesma taxa será usada para todas as saídas.

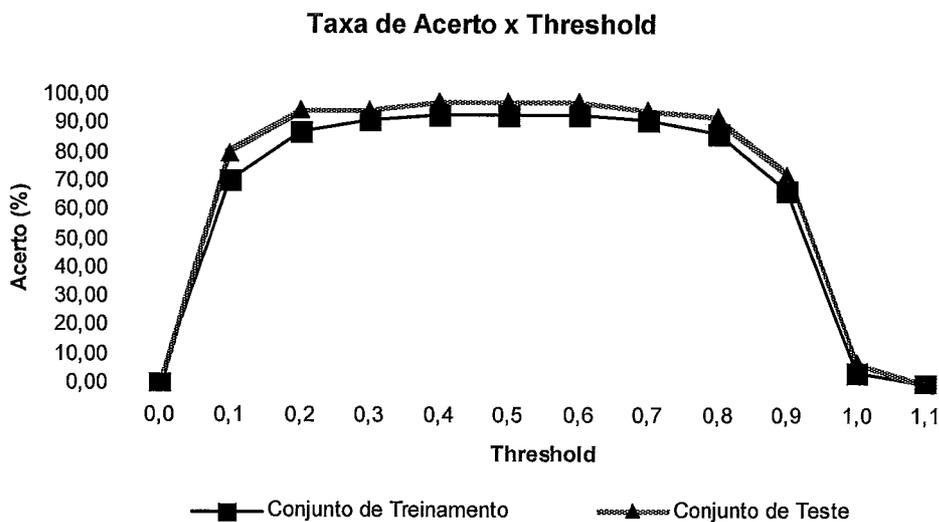


Gráfico 6.7. Acerto na classificação em função do *threshold*.

A maior taxa de acerto na classificação – 97,50% – foi obtida para valores de *threshold* entre 0,4 e 0,6. A figura 6.1 mostra a única CDF que não foi classificada corretamente pela rede com as saídas discretizadas com o *threshold* de 0,5. Originalmente, a carta foi classificada como Pancada de Fluido pelo especialista. Trata-se de uma pancada leve e, talvez por isso, a rede tenha confundido a classificação da carta.

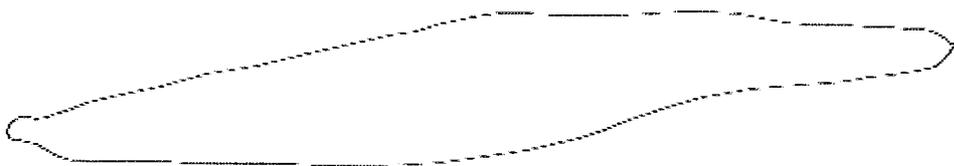


Figura 6.1. CDF classificada incorretamente.

## 6.5 MLP x Outros Tipos de RNA

### 6.5.1 Uma rede com $N$ saídas x $N$ redes com uma saída

Conforme apresentado anteriormente, o acervo de cartas classificadas é pequeno e, atualmente, não há exemplo de cartas de vários tipos de falhas que podem ocorrer no sistema de bombeio mecânico (SBM). Espera-se que novos tipos de cartas sejam adicionados a esse acervo com o tempo. Por isso, uma solução mais escalável seria ter um conjunto de redes neurais, cada uma especializada em uma situação operacional diferente, ao invés de se ter uma rede para todas as situações. Isto significa que, diante de  $N$  situações operacionais diferentes,  $N$  redes neurais com uma saída seria uma melhor solução que uma rede neural com  $N$  saídas.

Para treinar uma rede especialista, não se pode utilizar a mesma amostra de dados utilizada nos testes anteriores, pois precisa-se de uma amostra que tenha um número proporcional de padrões que apresentam uma determinada situação operacional e de padrões que não apresentam essa situação. Por isso, para treinar a rede especializada em cartas Normais, por exemplo, a amostra de dados foi composta pelas 100 cartas Normais da amostra original e por 100 cartas Não-Normais (40 cartas com Pancadas de Fluido, 40 cartas com Interferência de Gás e 20 cartas com outras classificações).

	Normal	Pancada de Fluido	Interferência de Gás	Outra Classificação	Média
MSE	0,0692	0,1396	0,0799	0,0337	0,0806
Desvio Padrão	0,0139	0,0210	0,0313	0,0064	0,0181

Tabela 6.1. Resultado das  $N$  redes de uma saída.

O resultado das  $N$  redes –  $N$  igual a 4 – de uma saída estão na tabela 6.1. As redes que apresentaram maior MSE foram as especialistas em Pancada de Fluido e Interferência de Gás. Em algumas situações, essas duas falhas de bombeio mecânico geram cartas semelhantes e, por essa razão, algumas cartas que indicam Pancada de Fluido devem ter sido classificadas como Interferência de Gás e vice-versa.

	4 redes de uma saída	Uma rede de 4 saídas
MSE	0,0806	0,0592
Desvio Padrão	0,0181	0,0186

Tabela 6.2.  $N$  redes de uma saída x Rede única de  $N$  saídas.

O MSE das quatro redes de uma saída, como pode ser observado na tabela 6.2, é maior do que o encontrado na rede única de quatro saídas e, portanto, não se justifica a criação de redes especialistas.

## 6.5.2 Redes Fuzzy ARTMAP

Conforme mencionado no Capítulo 3, Carpenter e Grossberg (1992) apresentaram as redes Fuzzy ARTMAP como redes que minimizam o erro de predição e maximizam a generalização simultânea e automaticamente. Essas características fizeram das redes Fuzzy ARTMAP uma abordagem interessante para o reconhecimento das cartas dinamométricas.

As redes Fuzzy ARTMAP têm apenas uma saída. Por isso, foram criadas 4 redes, uma para cada tipo de carta.

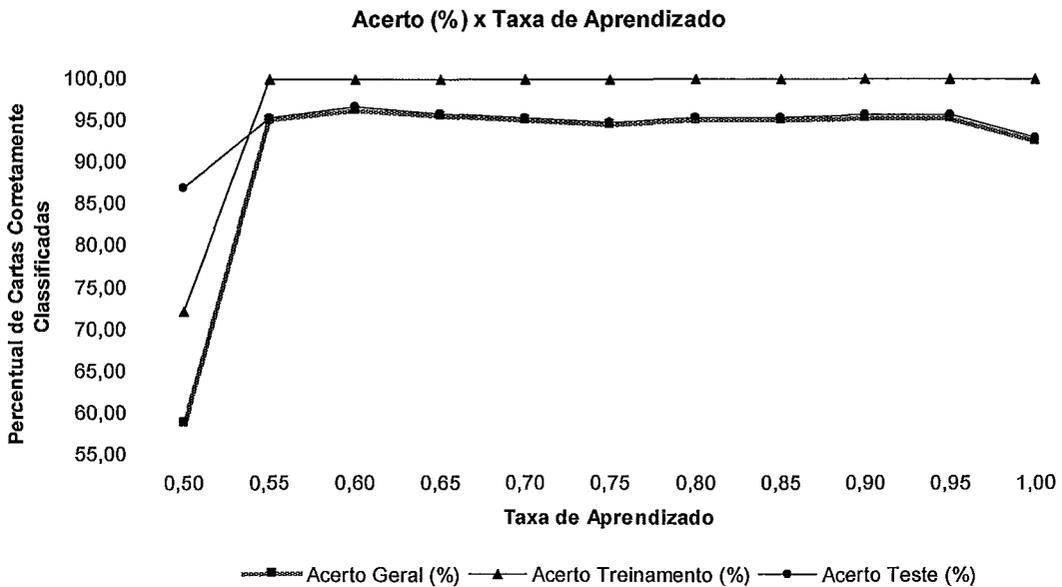
No primeiro teste realizado com as redes Fuzzy ARTMAP, utilizou-se a configuração padrão do *toolbox* ARTMAP<sup>11</sup> e a mesma amostra de dados com os 20 atributos selecionados pelo método Alternado. A taxa percentual de acerto foi de 92,75%, isto é, 92,75% das cartas foram classificadas corretamente.

A taxa de aprendizado padrão da rede Fuzzy ARTMAP é igual a 1 (um). Realizaram-se, também, testes com outras taxas de aprendizado para verificar o comportamento da rede. Como a iniciação dos pesos na rede Fuzzy ARTMAP não é aleatória, somente foi realizado um teste para cada valor de taxa de aprendizado (0,50; 0,55; 0,60; ...; 1,00).

O gráfico 6.8 mostra a taxa percentual de acerto na classificação em função da taxa de aprendizado. A taxa de acerto no conjunto de teste varia entre 94,75% e 96,50% para taxas de aprendizado entre 0,55 e 0,95, valores maiores do que o obtido com a taxa padrão de aprendizado. A taxa percentual de acerto se reduz abruptamente quando a taxa de aprendizado é igual a 0,50. A maior taxa de acerto – 96,50% – foi obtida com a taxa de aprendizado de 0,60. A tabela com os valores do gráfico se encontra no Apêndice A.

---

<sup>11</sup> <http://www.mathworks.com/matlabcentral/fileexchange/4306>



**Gráfico 6.8.** Acerto da rede Fuzzy ARTMAP em função da taxa de aprendizagem.

O acerto da rede Fuzzy não poderá ser melhorado – exceto se for alterado algum parâmetro de configuração da rede como, por exemplo, a taxa de vigilância – porque a iniciação dos seus pesos não é aleatória.

As redes Fuzzy ARTMAP oferecerem uma grande vantagem sobre as demais redes, visto que não é necessário se preocupar com a topologia da rede. Embora tenham apresentado um bom desempenho na classificação das cartas dinamométricas, o resultado obtido com as redes Fuzzy ARTMAP ainda é inferior àquele apresentado pela rede neural MLP com quatro neurônios na camada escondida. Por essa razão, elas não serão implementadas neste trabalho.

### 6.5.3 Mapas de Kohonen

Os mapas de Kohonen, conforme explicado no Capítulo 3, são redes não-supervisionadas. Embora, normalmente, eles sejam utilizados para fazer o agrupamento – e não a classificação! – de um conjunto de dados, o presente trabalho propôs a seguinte abordagem: utilizar um mapa de Kohonen obtido após o treinamento com um conjunto de padrões para classificar novas amostras.

Para verificar se o mapa obtido durante o treinamento poderá ser usado para classificar novos padrões, é necessário que se apresente ao mapa um conjunto de padrões diferentes daqueles utilizados no processo de aprendizagem: o conjunto de

teste. Para cada novo padrão, deve-se verificar se ele ativa os mesmos neurônios que foram ativados por padrões do conjunto de treinamento que pertenciam à sua classe (grupo). Dessa forma, podem ser obtidas outras métricas além das apresentadas no Capítulo 3, tais como: o percentual de acertos ( $a$ ), o percentual de erros ( $e$ ) e o percentual de ocorrências de fronteira ( $f$ ). A explicação dessas métricas pode ser vista no Apêndice B.

Diversos tipos de mapa de Kohonen foram testados para o agrupamento e, posteriormente, para a classificação das cartas. O mapa de Kohonen que apresentou o melhor desempenho – 84,57% de acerto na classificação do conjunto de teste – foi um mapa de reticulado hexagonal, forma planar e de tamanho 10 x 15 neurônios. A descrição completa desse e dos demais testes com mapas de Kohonen está no Apêndice A.

O mapa de Kohonen que obteve a maior taxa de acerto na classificação de padrões do conjunto de teste pode ser visto na figura 6.2. Cada neurônio herdou o rótulo do padrão que o ativou. Caso um neurônio tenha sido ativado mais de uma vez, por padrões de diferentes classes, terá herdado o rótulo da classe com maior número de representantes nas ativações daquele neurônio.

Pode-se ver, pela figura 6.2, que a maior parte das cartas Normais encontram-se no lado esquerdo do mapa. As cartas com Interferência de Gás predominam a parte superior da região direita e as cartas com Pancada de Fluido a parte central inferior. As cartas com Outras Classificações encontram-se distribuídas no mapa, já que, de fato, não são cartas pertencente a uma única classe. As unidades do mapa que não têm rótulo representam a fronteira entre as classes (grupos).

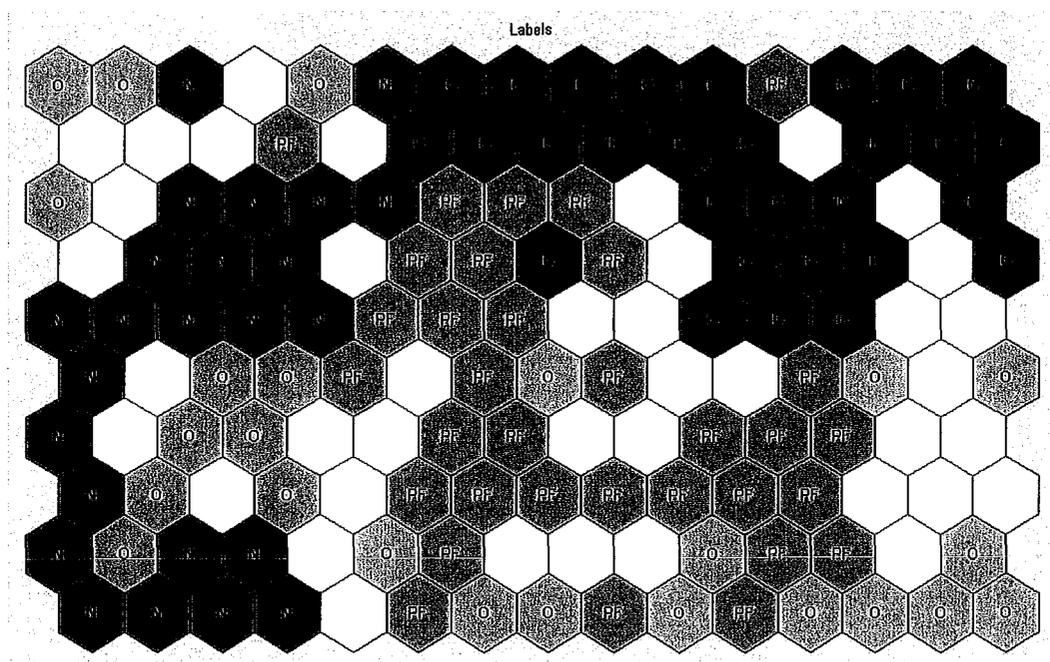


Figura 6.2. Mapa de Kohonen com neurônios ativados destacados.

A figura 6.3 mostra a Matriz-U, que representa a distância entre as unidades do mapa, inserindo uma célula adicional entre cada par de neurônios. Como se pode perceber, a identificação visual dos grupos de cartas através dessa matriz nem sempre é uma tarefa simples.

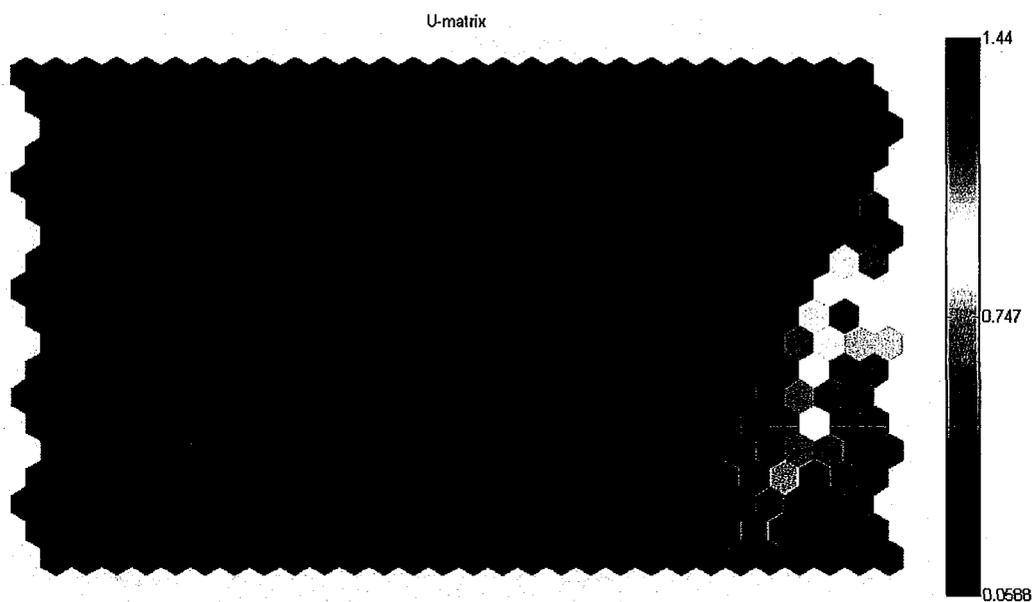


Figura 6.3. Matriz-U: indica distância entre as unidades do mapa.

Tal como esperado, o resultado obtido com o mapa de Kohonen não foi superior ao obtido com as redes do tipo MLP, confirmando que a sua aplicação é indicada

quando se deseja descobrir grupos em um conjunto de padrões e não para classificar novos padrões.

## 6.6 Comparação com os Trabalhos Relacionados

Como foi apresentado, a rede neural MLP com 20 entradas e 4 saídas apresentou o melhor resultado na classificação das cartas do conjunto de teste: 97,50% das cartas foram classificadas corretamente. A tabela 6.3 apresenta a comparação entre rede neural utilizada no presente trabalho – aqui denominado Cianni (2009) – e as redes neurais dos trabalhos relacionados (apresentados no capítulo 4). O caractere “-” é apresentado quando não há informação sobre o atributo que representa a coluna da tabela ou quando o atributo não se aplica ao trabalho.

Trabalhos	Entradas	Camada Escondida	Saídas	Pesos	Ciclos de Treinamento
Cianni (2009)	20	4	4	96	10
Martinez <i>et al.</i> (1993)	30	8	21	408	10000
Xu <i>et al.</i> (2007)	216	-	5	1080	100
Nazi <i>et al.</i> (1994)	80	60	11	5460	20000
Rogers <i>et al.</i> (1990)	19	279	2	5859	3459

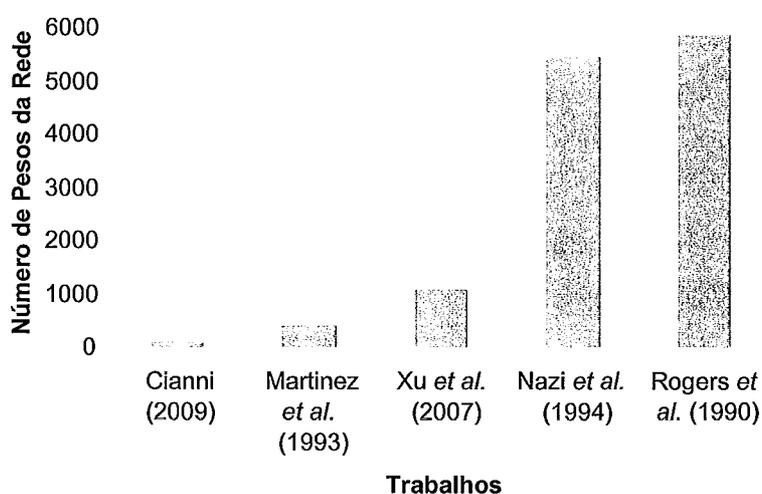
Tabela 6.3. Comparação entre as redes neurais de cada trabalho.

A quantidade de ciclos utilizados para o treinamento da rede neural em (Cianni, 2009) é bem menor que nos demais trabalhos porque foi adotado o método da Parada Antecipada para interromper o processo de aprendizagem quando houvesse indícios de supertreinamento (*overfitting*). Além disso, o algoritmo de treinamento utilizado foi o Levenberg-Marquardt, que converge em um número menor de ciclos. Os trabalhos relacionados que utilizaram mais de 1.000 ciclos de treinamento podem ter supertreinado a rede.

O tamanho da rede neural apresentada em (Cianni, 2009), como pode ser visto mais claramente no gráfico 6.8, também é inferior ao tamanho das demais redes.

O tamanho reduzido da rede e do número de ciclos de treinamento utilizados em (Cianni, 2009) diminuem o risco da rede ser supertreinada, isto é, da rede decorar os padrões do conjunto de treinamento e perder sua capacidade de generalização para classificar cartas que não foram utilizados no conjunto de treinamento.

### Tamanho da Rede Neural



**Gráfico 6.8.** Comparação entre o tamanho das redes de cada trabalho.

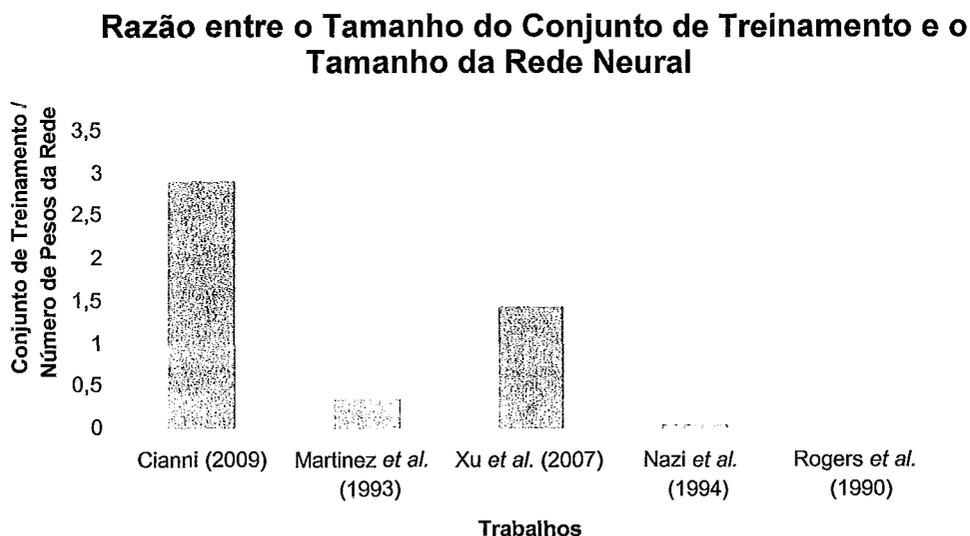
A tabela 6.4 apresenta a comparação entre as amostras de dados utilizadas neste e nos demais trabalhos. Nela, pode-se ver que apenas (Cianni, 2009) separa um conjunto de validação para verificar o problema de supertreinamento da rede neural.

Trabalhos	Conjunto de Treinamento	Conjunto de Validação	Conjunto de Teste	Total	Acerto (%) na Classificação	Dados igualmente distribuídos
Cianni (2009)	280	80	40	400	97,50	Sim
Martinez et al. (1993)	140	0	50	190	95,00	-
Xu et al. (2007)	1550	0	6337	7887	99,97	Não
Nazi et al. (1994)	176	0	20	196	100,00	-
Rogers et al. (1990)	15	0	0	15	-	Não

**Tabela 6.4.** Comparação entre a amostra de dados de cada trabalho.

Além disso, somente Cianni (2009) utilizou uma amostra de cartas igualmente distribuídas entre das classes. Os demais trabalhos não se preocuparam em utilizar ou em mencionar que utilizaram um conjunto de treinamento proporcionalmente distribuído. Xu et al. (2007), por exemplo, apresentam um conjunto de cartas em que duas das cinco classes representam cerca de 83,85% de toda a amostra.

Outra questão que deve ser analisada é a relação entre o tamanho do conjunto de treinamento e o tamanho da rede neural. Quanto maior a rede, maior deve ser o número de padrões do conjunto de treinamento. O que se observa, no entanto, é que grandes redes apresentadas nos trabalhos relacionados são treinadas com pequenas amostras. Este trabalho (Cianni, 2009) é o que apresenta a maior razão entre o número de cartas dinâmométricas do conjunto de treinamento e o número de pesos da rede neural, como pode ser visto no gráfico 6.9.



**Gráfico 6.9.** Relação entre o conjunto de treinamento e o tamanho da rede.

O trabalho de Alegre *et al.* (1993), conforme explicado no capítulo 4, utiliza conjunto de *templates* de cartas definidos previamente por um especialista ao invés de cartas reais para treinamento. Também é papel do especialista definir os pesos da rede e ajustá-los quando não apresentam bons resultados. Por isso, esse trabalho não entrou na comparação acima.

## 7 IFBM – Módulo de Identificação de Falhas em Sistemas de Bombeio Mecânico a partir de Cartas Dinamométricas de Fundo

O produto final deste trabalho é o desenvolvimento do Módulo de Identificação de Falhas em Sistemas de Bombeio Mecânico a partir de Cartas Dinamométricas de Fundo (IFBM). Esse módulo foi implementado dentro de um sistema de análise de dados de produção da Petrobras, denominado SOLAR.

Conforme explicado no decorrer desta dissertação, para identificar a falha, o módulo faz uso de uma rede neural para classificar a carta de acordo com a situação operacional que a sua forma representa. A rede neural utilizada no trabalho foi treinada previamente no Matlab. Para incorporar a rede treinada ao SOLAR, optou-se pelo desenvolvimento de um módulo de redes neurais artificiais.

Esta seção apresenta o SOLAR, o módulo de Redes Neurais Artificiais (RNA), a arquitetura e a interface do módulo IFBM.

### 7.1 SOLAR - Sistema de Otimização e Análise Remota de Dados de Produção

O SOLAR – Sistema de Otimização e Análise Remota de Dados de Produção – é uma solução interna da Petrobras para o gerenciamento integrado da produção através da monitoração e análise de dados em tempo real.

A figura 7.1 mostra a interface do SOLAR com alguns de seus módulos e ferramentas sendo utilizados.

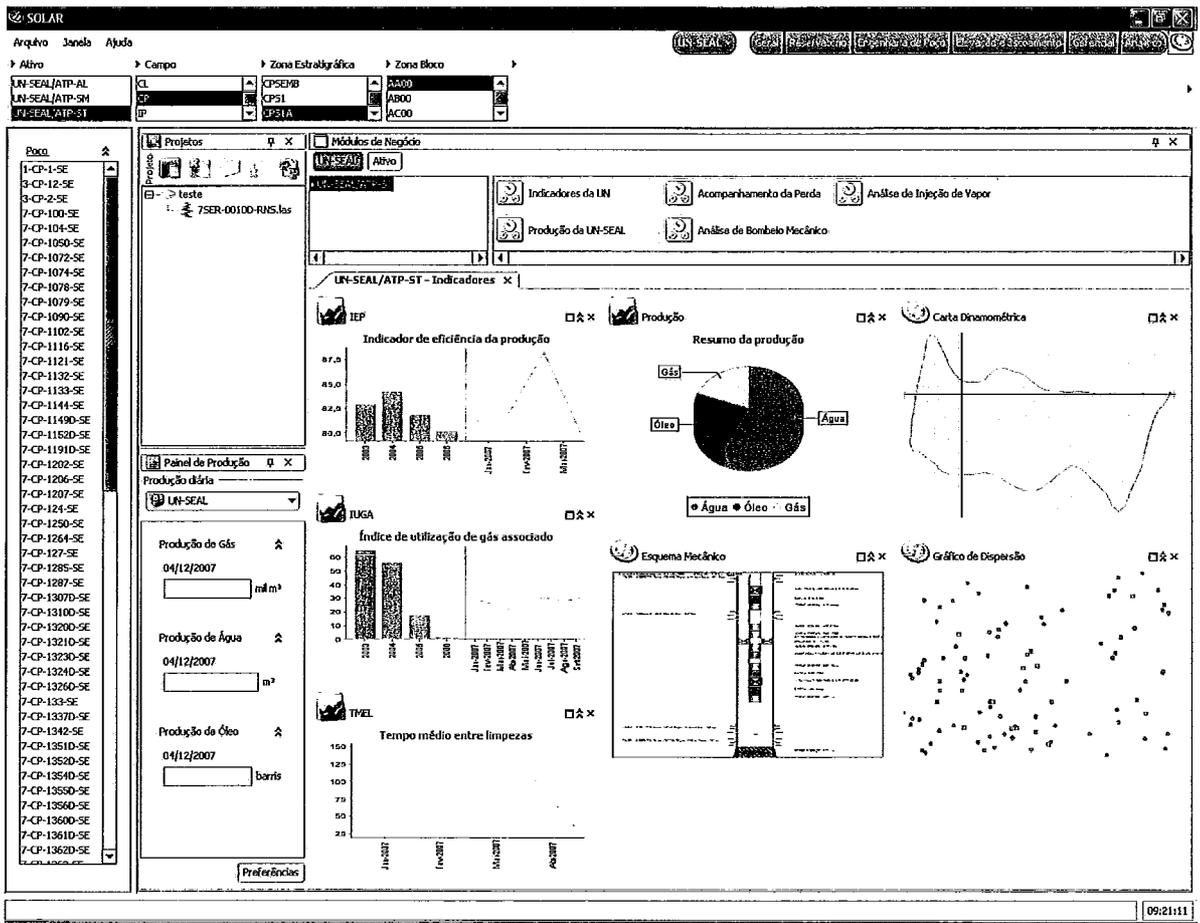


Figura 7.1. Interface do SOLAR.

O SOLAR permite que informações de diferentes áreas da produção – reservatório, elevação artificial e escoamento, engenharia de poços e automação – sejam combinadas e representadas em uma interface comum, e, por isso, pode ser usado como uma ferramenta colaborativa em tomadas de decisões multidisciplinares.

## 7.2 Módulo de Redes Neurais Artificiais (RNA)

O objetivo do módulo de Redes Neurais Artificiais (RNA) é incorporar a rede treinada no Matlab, ou em outro programa, ao IFBM. O treinamento da rede, portanto, não faz parte do escopo inicial desse módulo.

O módulo RNA foi desenvolvido fora do SOLAR para que ele possa ser usado e estendido por outros sistemas. O funcionamento básico do RNA consiste em criar uma rede neural em memória de forma que as suas saídas sejam equivalentes às saídas do programa no qual ela foi treinada.

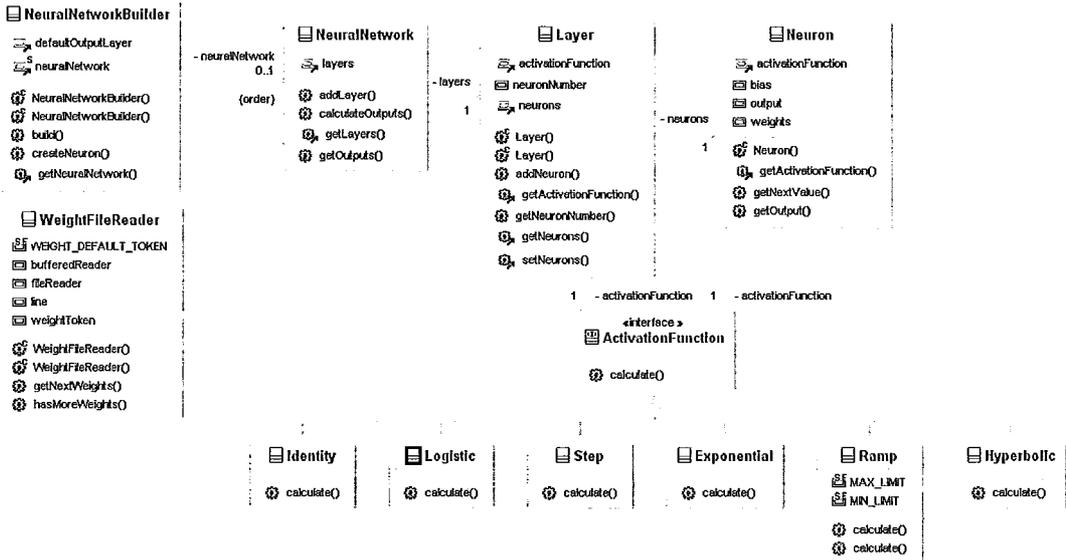


Figura 7.2. Diagrama de Classes do Módulo RNA.

A figura 7.2 apresenta o diagrama de classes do módulo RNA.

A rede neural é criada a partir de um arquivo que deve conter os biases e pesos de cada neurônio e a partir da definição das camadas da rede. A seguir, será apresentado um exemplo de uso do módulo RNA para tornar mais claro o seu entendimento.

### 7.2.1 Exemplo do AND de Três Entradas

Para mostrar o funcionamento do módulo, criou-se, no Matlab, uma rede neural com dois neurônios na camada intermediária e um neurônio na camada de saída. Em seguida, treinou-se a rede para aprender o funcionamento da função AND de três entradas. A figura 7.3 apresenta o código para criação, treinamento e simulação da rede neural gerada no Matlab.

```

% Inputs - entradas
P = [0 0 0; 0 0 1; 0 1 0; 0 1 1;
     1 0 0; 1 0 1; 1 1 0; 1 1 1]';

% Targets - saídas desejadas
T = [0 0 0 0 0 0 0 1];

% Define a rede
hiddenLayerSize = 2;
net = newfit(P, T, hiddenLayerSize);

% Elimina pré-processamento
net.inputs{1}.processFcns = {};

% Treina a rede
[net, tr] = train(net, P, T);

% Elimina pós-processamento
net.outputs{2}.processFcns = {};

% Simula a rede (Y = output = saída da rede)
Y = sim(net, P)';

```

Figura 7.3. Código para criação, treinamento e simulação da rede no Matlab.

A rede neural *net*, criada no Matlab, pode ser visualizada na figura 7.4. Para que essa mesma rede seja recriada pelo módulo RNA em memória, é necessário criar um arquivo com o bias e os pesos de cada neurônio.

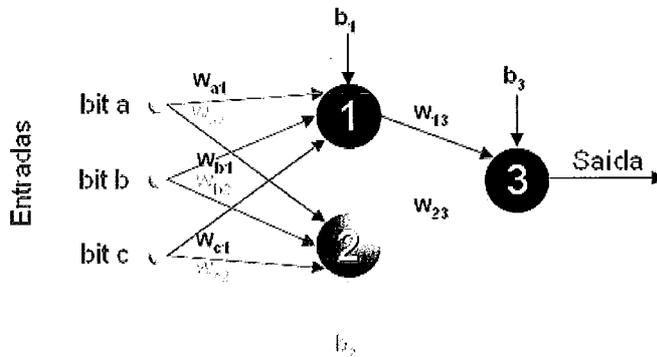


Figura 7.4. Rede criada no Matlab e recriada no módulo RNA.

A figura 7.5 apresenta o formato do arquivo esperado pelo módulo RNA. Para cada neurônio da rede neural, deve-se criar, no arquivo, uma linha com o bias e os pesos do neurônio, separados por ponto-e-vírgula. O número de linhas do arquivo é, portanto, igual ao número de neurônios da rede neural.

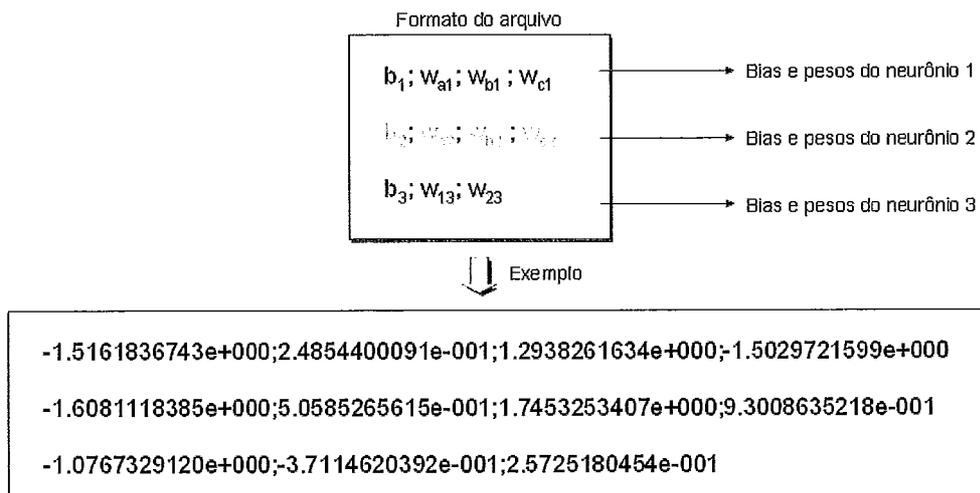


Figura 7.5. Exemplo de arquivo esperado pelo módulo RNA.

Criado o arquivo com os biases e pesos da rede neural, é necessário definir as demais características da rede, que não podem ser inferidas a partir do arquivo no módulo RNA.

A figura 7.6 mostra um exemplo de código que utiliza o módulo RNA para recriar a rede neural da figura 7.4, treinada no Matlab. A primeira linha informa o nome do arquivo de pesos da rede, como o que foi apresentado na figura 7.5. Em seguida, cria-se uma camada intermediária com dois neurônios. A função de ativação *tangente hiperbólica* (classe *Hyperbolic*) atribuída à camada intermediária é a função utilizada pelos neurônios dessa camada. A camada intermediária e o arquivo são usados para criar a rede. Quando a camada de saída não é especificada, utiliza-se a função *linear* (classe *Identity*) como função de ativação para os neurônios dessa camada.

A rede neural *neuralNetwork*, criada pela classe *NeuralNetworkBuilder* (figura 7.6), equivale à rede *net* criada no Matlab (figura 7.3). A saída gerada pela rede criada pelo módulo RNA aparece na figura 7.7. Até a décima casa decimal, a saída é idêntica àquela obtida na simulação da rede no Matlab, o suficiente para se obter o mesmo resultado final.

```

// Arquivo de pesos da rede
File file = new File("pesos.dat");

// Camada intermediária (Hidden Layer) da rede neural com 2 neurónios e função de ativação TANH
Layer layer = new Layer(2, new Hyperbolic());

// Constrói a rede
NeuralNetworkBuilder builder = new NeuralNetworkBuilder(file, layer);
NeuralNetwork neuralNetwork = builder.getNeuralNetwork();

// Imprime as saídas da rede para um conjunto de entradas informadas em um arquivo.
System.out.println("Input          Fuzzy Output          Crisp Output");
InputFileReader inputFileReader = new InputFileReader(new File("ANDExampleInputs.dat"));
while (inputFileReader.hasMoreInputs()) {
    List<Double> inputs = inputFileReader.getNextInputs();
    // Obtém as saídas da rede neural para as entradas fornecidas
    List<Double> outputs = neuralNetwork.getOutputs(inputs);
    List<Integer> crispOutputs = new ArrayList<Integer>();
    for (Double output : outputs) {
        double threshold = -0.507; //Limite definido arbitrariamente.
        if (output > threshold) {
            crispOutputs.add(1);
        } else {
            crispOutputs.add(0);
        }
    }
    System.out.println(inputs + "          " + outputs + "          " + crispOutputs);
}

```

Figura 7.6. Exemplo de código do módulo RNA para recriar a rede treinada.

Input	Fuzzy Output	Crisp Output
[0.0, 0.0, 0.0]	[-0.9771335985396372]	[0]
[0.0, 0.0, 1.0]	[-0.8591921212892277]	[0]
[0.0, 1.0, 0.0]	[-0.9604609711546647]	[0]
[0.0, 1.0, 1.0]	[-0.5255835400763547]	[0]
[1.0, 0.0, 0.0]	[-0.9662252629724928]	[0]
[1.0, 0.0, 1.0]	[-0.7523456480149883]	[0]
[1.0, 1.0, 0.0]	[-0.9405921407123324]	[0]
[1.0, 1.0, 1.0]	[-0.5063454930402838]	[1]

Figura 7.7. Saída da rede neural criada pelo módulo RNA na figura 7.6.

### 7.3 A Arquitetura do IFBM

O IFBM foi implementado no SOLAR e, portanto, segue a mesma arquitetura desse sistema, apresentada na figura 7.8. Na primeira camada, estão os sistemas de supervisão e aquisição de dados, conhecidos como Scada (*Supervisory Control And Data Acquisition*). Esses sistema recuperam os dados da automação que, posteriormente, são registrados em uma base histórica apropriada, conhecida como PI (*Plant Information*). As cartas dinamométricas são exemplos de dados vindos da automação e registrados no PI. Dados armazenados no PI são, por vezes, registrados em

uma base de dados relacional e podem ser acessados diretamente ou através de uma camada serviços.

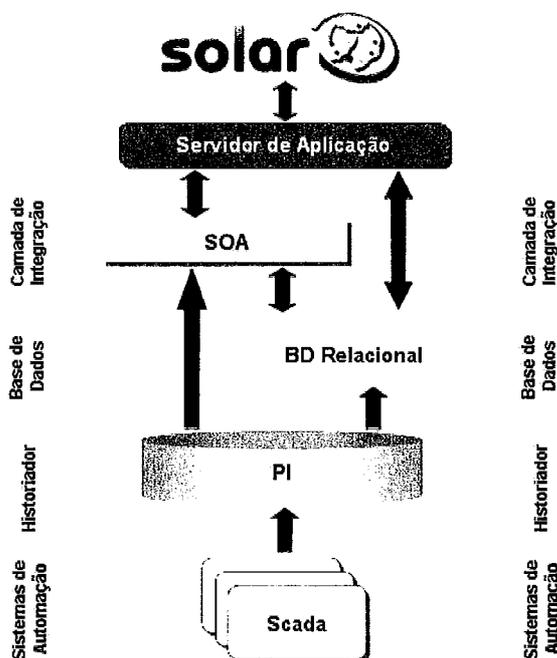


Figura 7.8. Arquitetura do SOLAR.

De uma forma mais simplificada, pode-se apresentar as interfaces do IFBM (figura 7.9) através dos seguintes passos: (1) o módulo recupera uma carta armazenada na base de dados; (2) repassa-a para o módulo RNA; (3) recupera a classificação informada pelo módulo RNA; (4) e, por fim, armazena a classificação da carta na base de dados.



Figura 7.9. Interfaces do módulo: visão geral.

O módulo está dividido em quatro etapas principais (apresentadas na figura 7.10):

1. Normalização – O primeiro passo é normalizar os pontos da carta, conforme explicado no capítulo 5.
2. Seleção das Coordenadas – A carta dinamométrica de fundo possui cerca de 100 pontos, cada um formado pelo par de coordenadas de carga vs. posição. Nesta etapa, são descartadas as 100 coordenadas de posição e selecionadas as 20 coordenadas de carga, pelo Método Alternado, que serão usadas para a classificação. Aqui, também são eliminadas as cartas que estão incompletas ou que só possuem cargas negativas (provavelmente, devido a algum problema no sensor).
3. Pré-Classificação – Alguns tipos de carta podem ser identificados, facilmente, sem o auxílio da rede neural. Isso reduz o número de classificações que a rede tem que aprender e, conseqüentemente, melhora o seu desempenho.

Um problema que pode ser pré-classificado, por exemplo, ocorre quando a haste do sistema de bombeio mecânico se parte. Essa situação é identificada quando a diferença entre as cargas máxima e mínima registradas na carta é inferior ao peso da coluna de hastes.

4. Classificação Final – Ocorre para as cartas que não foram classificadas na Pré-Classificação. Consiste em submeter as cartas à rede neural para se obter a sua classificação.



Figura 7.10. Etapas realizadas pelo módulo.

## 7.4 A Interface do IFBM

A interface desenvolvida para o módulo IFBM (figura 7.11) permite ao usuário visualizar as cartas dinamométricas de um poço cujo método de elevação seja Bombeio Mecânico. Ao selecionar uma das cartas exibidas, o usuário a vê destacada em um gráfico maior na parte inferior da tela e a sua classificação ao lado.

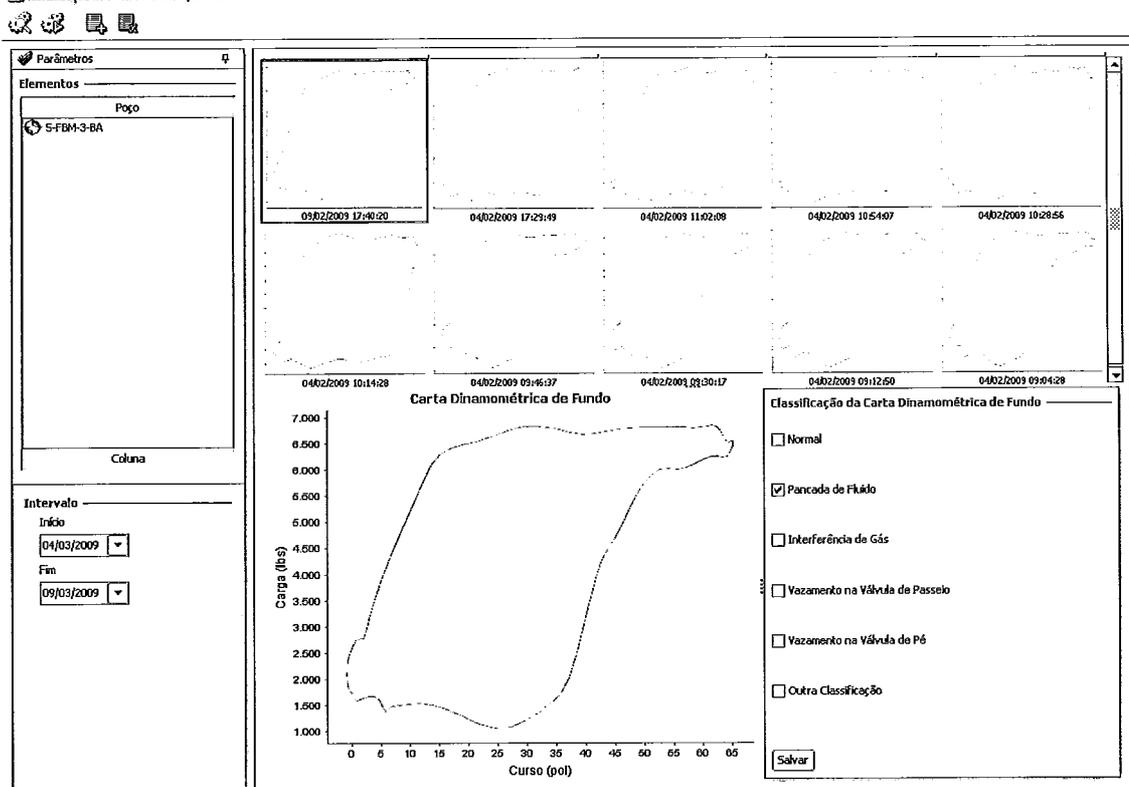


Figura 7.11. Interface do módulo.

Caso o usuário não concorde com a classificação realizada pela rede neural, ele pode atribuir uma nova classificação à carta, o que serve para aumentar e corrigir a base atual de cartas classificadas.

O Apêndice C apresenta uma explicação mais detalhada sobre a interface do SOLAR e sobre a interface do IFBM.

## 8 Conclusões

Este trabalho apresentou uma solução para o diagnóstico de falhas em bombeio mecânico a partir da classificação de cartas dinamométricas de fundo realizada por uma rede neural supervisionada.

Essa solução difere das demais existentes na literatura por descartar todas as coordenadas de posição e utilizar somente um quinto das coordenadas de carga da carta dinamométrica para o seu reconhecimento. As coordenadas de posição foram descartadas após se mostrar que apenas as coordenadas de carga podem ser suficientes para se identificar uma falha no sistema de bombeio.

Técnicas para evitar o supertreinamento (*overfitting*) foram aplicadas à rede proposta. Dentre elas, estão a utilização de um conjunto de validação para determinar o número ideal de ciclos de treinamento e a redução do número de entradas e pesos da rede. Realizou-se também a validação cruzada para garantir que o conjunto de treinamento escolhido fosse aquele que oferecesse melhor generalização na classificação das cartas do conjunto de teste. Apesar de se tratarem de assuntos comuns na literatura sobre redes neurais, nenhum dos trabalhos relacionados a este menciona o uso de tais técnicas, conforme mencionado no capítulo 6.

Atualmente, o acervo de cartas classificadas utilizado neste trabalho somente contém um número suficiente de exemplares de três classes: Normal, Pancada de Fluido e Interferência de Gás. As cartas que não pertenciam a nenhuma dessas classes compuseram um quarto grupo de cartas, denominado Outra Classificação. A rede neural foi treinada com cartas igualmente distribuídas nessas classes e, portanto, somente é capaz de classificar uma carta dinamométrica em uma dessas quatro classes.

Um módulo de Identificação de Falhas em Sistemas de Bombeio Mecânico a partir de Cartas Dinamométricas de Fundo, denominado IFBM, foi implementado para facilitar a classificação das cartas. Esse módulo faz parte do Sistema de Otimização e Análise Remota de Dados de Produção (SOLAR) desenvolvido na Petrobras.

Para incorporar a rede treinada no Matlab ao IFBM, foi necessária a implementação de um módulo de Redes Neurais Artificiais (RNA). O módulo RNA foi desenvolvido fora do SOLAR para que possa ser usado por outros sistemas. Nesse

módulo, uma rede neural é criada a partir da definição de suas camadas escondidas e a partir de um arquivo que contém os pesos de suas conexões. As informações que compõem esse arquivo podem ser obtidas após o treinamento da rede no Matlab, no Weka<sup>12</sup> ou em qualquer programa capaz de treinar uma rede neural.

A rede neural proposta neste trabalho classificou corretamente 97,50% das 40 cartas do conjunto de teste. Apesar do excelente desempenho obtido na classificação das cartas do conjunto de teste, acredita-se que esse resultado somente será refletido na prática quando o acervo de cartas classificadas incluir outros tipos de falhas – tais como Vazamento na Válvula de Passeio e Vazamento na Válvula de Pé – e apresentar um número de cartas representativo o suficiente para se treinar a rede neural.

O módulo de diagnóstico de falhas em bombeio mecânico foi implementado de forma que o usuário possa classificar manualmente novas cartas e possa corrigir a classificação realizada pela rede neural, aumentando o acervo original de cartas classificadas. Dessa forma, em um futuro próximo, uma nova rede poderá ser treinada com mais classes de cartas e com mais cartas em cada classe.

## 8.1 Trabalhos Futuros

Como trabalhos futuros, sugere-se realizar uma investigação sobre as informações que auxiliam no diagnóstico de cada falha possível no sistema de bombeio mecânico e avaliar a utilização dessas informações na rede neural proposta. Deve-se avaliar também o uso dessas informações em outras redes neurais que possam complementar a classificação obtida neste trabalho, isto é, estudar a criação de um comitê de redes neurais para identificar a falha no sistema de bombeio.

Ainda pensando em melhorar o resultado da classificação, pode-se aplicar a análise de Curvas ROC (*Receiver Operating Characteristic*) para definir o *threshold* de cada saída da rede de forma a maximizar o seu desempenho.

O módulo RNA pode ser ampliado para contemplar outros tipos de redes neurais, como, por exemplo, as Redes Fuzzy ARTMAP e os Mapas de Kohonen, estudados neste trabalho. Pode-se avaliar, também, o ganho em se construir proxies para acessar as estruturas do Matlab diretamente do módulo RNA.

---

<sup>12</sup> <http://www.cs.waikato.ac.nz/ml/weka/>

Para o uso mais eficiente do módulo IFBM, pretende-se criar procedimentos automáticos que classifiquem um conjunto de cartas, uma de cada poço, em *background* e em intervalos de tempos pré-definidos, para que o usuário não precise disparar o processo de classificação da rede neural. Deseja-se também desenvolver uma interface que permita ao usuário visualizar rapidamente os poços que apresentam falha no bombeio mecânico detectada a partir da classificação da carta.

Além disso, conforme já mencionado, espera-se treinar a rede novamente quando houver um acervo maior de cartas classificadas (inclusive com novas classes de cartas e classes sobrepostas), ampliando, assim, a aplicação dessa ferramenta.

# Referências Bibliográficas

- Alegre, L., Rocha, A. F., Morooka, C. K., **Intelligent Approach of Rod Pumping Problems**, SPE 26253, 1993.
- Alegre, L., Rocha, A. F., Morooka, C. K., **Intelligent Diagnosis of Rod Pumping Problems**, SPE 26516, 1993.
- Alilat, F., Loumi, S., Merrad, H., Sansal, B. **Modified Fuzzy ARTMAP and Supervised Fuzzy ART: Comparative Study with Multispectral Classification**, *International Journal of Computer Science*. (1)4:1306-4428, 2006.
- Barreto Filho, M.A., **Geração de carta dinamométrica de fundo para diagnóstico de bombeio mecânico em poços de petróleo**, Dissertação de Mestrado da Universidade Estadual de Campinas, 1993.
- Berry, M. J. A., Linoff, G. S. **Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management**, 2nd Edition, Wiley Computer Publishing, 2004.
- Bezerra, Murilo Valença, **Avaliação de Métodos de Elevação Artificial de Petróleo utilizando Conjuntos Nebulosos**, Dissertação de Mestrado, UNICAMP, 2002.
- Bishop, C. M., **Neural Networks for Pattern Recognition**, 1995, Oxford University Press, Oxford, UK.
- Braga, A. C. da S., **Curvas ROC: Aspectos Funcionais e Aplicações**, Tese de doutorado, Universidade do Minho, dezembro, 2000.
- Brown, K.E., **Overview of Artificial Lift Systems**, Journal of Petroleum Technology, 1982, SPE 9979, pp. 2384-2396
- Câmara, G. Oliveira M. C. M., **Rede de Pulso Acoplado Seqüencial para Reconhecimento de Padrões Invariantes**, Departamento de Ciência da Computação - Universidade Federal de Minas Gerais, Campus Pampulha - Belo Horizonte, Agosto 2003. Encontrado em: <http://homepages.dcc.ufmg.br/~gcamarac/visao/visaoslides.pdf>.

- Carpenter, G. A., Grossberg, S., Markuzon, N. and Rosen, D. B., **Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps**, IEEE Trans. Neural Networks, 3(5):698-713, 1992.
- Carvalho, L. A. V., **Datamining: A mineração de dados no marketing, medicina, economia, engenharia e administração**, 2ª Edição, Editora Érica, 2001.
- Cerqueira, J. F., Corrêa, J. F., Lepkison, H., Bittencourt, A. C., Schnitman, L. Jesus, A. B., **Development of an Intelligent Distributed Management System for Automated Wells (SGPA)**, 2002, SPE 77609.
- Ceylan S. E., **Design and Economic Evaluation of Sucker Rod and Electrical Submersible Pumps: Oil Wells in a Field**, Turkey. Msc Thesis, 2004.
- Corrêa, J. F., **Sistema Inteligente para Aplicações de Soluções ao Bombeamento Mecânico de Petróleo**, dissertação de mestrado, 1995, UNICAMP, Campinas.
- Corrêa, J. F. S., **CartaPad – cartas dinamométricas padrões**. Apresentado no II Seminário de Engenharia de Poços da Petrobras, 1998, Rio de Janeiro.
- Corrêa, J. F., Lepkison, H., Bittencourt, A., **Intelligent Distributed Management System for Automated Wells (SGPA) Experience and Results**, 2003, SPE 84225.
- Costa, Rutácio O., **Curso de Bombeio Mecânico**, Petrobras, 2005.
- Cybenko, G., **Continuous valued neural networks with two hidden layers are sufficient**, 1988, Technical report, Department of Computer Science, Tufts University.
- Cybenko, G., **Approximation by superpositions of a Sigmoidal Function**, 1989, Mathematics of Control, Signals and Systems, 2, 303-314.
- Demuth, H., Beale, M., Hagan, M., **Neural Network Toolbox™ 6 User's Guide**, The MathWorks, Inc., 2008.
- Derek, H. J., Jennings, J.W., Morgan S. M., **Sucker Rod Pumping Unit Diagnostics Using an Expert System**, Journal of Petroleum Technology, 1988, SPE 17318.

- Dickinson, R. R., Jennings, J. W., **Use of Pattern-Recognition Techniques in Analyzing Downhole Dynamometer Cards**, 1990, SPE 17313.
- Foley, W.L., Svinos, J.G., **EXPROD: Expert Advisor Program for Rod Pumping**, Journal of Petroleum Technology, 1987, SPE 16920.
- Gibbs, S. G., Neely, A. B., **Computer Diagnosis of Down-Hole Conditions In Sucker Rod Pumping Wells**, SPE 01165, 1966.
- Gibbs, S. G., **A Method of Determining Sucker Rod Pump Performance**, U. S. Patent N° 3,343,409, Setembro, 1967
- Gipson, F. W. - Swaim, H. W.: **The Beam Pumping Design Chain**. Proc. 31st Annual Southwestern Petroleum Short Course. Lubbock Texas, 1984. 296-376.
- Haykin, S. **Neural Networks: A Comprehensive Foundation**, 2nd Edition, Prentice Hall, 1998.
- Keating, J.F., Laine, R.E. and Jennings, J.W., **Pattern Recognition Applied to Dynamometer Cards**, 1989, SPE 19394, SPE Annual Technical Conference and Exhibition, 8-11 October 1989, San Antonio, Texas
- Keating, J.F., Laine, R.E. and Jennings, J.W., **Application of a Pattern-Matching Expert System to Sucker-Rod, Dynamometer-Card Pattern Recognition**, 1991, SPE 21666.
- Lippmann, R. P., **An introduction to computing with neural nets**, 1988, In Artificial Neural Networks: theoretical Concepts, V. Vemuri, Ed. Computer Society Press Technology Series Neural Networks. IEEE Computer Society Press, Los Alamitos, CA, 36-54.
- Martinez, E. R., Moreno W. J., Castillo, V. J., Moreno, J. A., **Rod pumping Expert System**, 1993, SPE 26246.
- Metz, C. E., **Statistical Analysis of ROC Data in Evaluating Diagnostic Performance**, *Multiple Regression Analysis: Applications in the Health Sciences*, número 13, editado por Donald E. Herbert and Raymond H. Myers. 365–384. American Institute of Physics, 1986.
- Minsky, M; Papert, S. **Perceptrons: An Introduction to Computational Geometry**. MIT Press, Cambridge, MA. 1969.

- Nascimento, J. M. A., **Simulador Computacional para Poços de Petróleo com Método de Elevação Artificial por Bombeio Mecânico**, Dissertação de Mestrado da Universidade Federal do Rio Grande do Norte, 2005.
- Nascimento, J. M. A., Alsina, P. J., Maitelli A. L., Costa, R. O., **Simulador Computacional para Poços de Petróleo com Método de Elevação Artificial por Bombeio Mecânico**, 3º Congresso Brasileiro de P&D em Petróleo e Gás, Salvador, 2005.
- Nazi G. M., Ashenayi K., Lea J. F., Kemp F., **Application of Artificial Neural Network to Pump Card Diagnosis**, Journal of Petroleum Technology, 1994, SPE 25420.
- Nepomuceno, A. M., **Uso de Rede Neural Artificial Não Supervisionada na Classificação de Dados de Radar na Banda-P para Mapeamento de Cobertura da Terra em Floresta Tropical**, Dissertação de Mestrado do Curso da Pós-Graduação em Sensoriamento Remoto, INPE, São José dos Campos, 2003. Disponível em: <http://mtc-m12.sid.inpe.br/col/sid.inpe.br/jeferson/2003/05.07.10.44/doc/publicacao.pdf>.
- Ranganathan, A. **The Levenberg-Marquardt Algorithm**, 2004. Disponível em: <http://www.cc.gatech.edu/~ananth/docs/lmtut.pdf>
- Reed, R. D., Marks, R. J. II. **Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks**, MIT Press, 1999.
- Rogers, J. D., Guffey, C. G., Oldham, W. J. B., **Artificial Neural Networks for Identification of beam Pump Dynamometer Load Cards**, 1990, SPE 20651.
- Schirmer, P., Gay, J. C., Toutain, P. **Use of Advanced Pattern-Recognition and Knowledge-Based System in Analyzing Dynamometer Cards**, 1991, SPE 22305.
- Schnitman, L., Albuquerque, G. S., Corrêa, J. F., Lepkison, H., Bittencourt, A. C. P., **Modeling and Implementation of a system for sucker rod downhole dynamometer card pattern recognition**, 2003, SPE 84140.
- Schnitman, L., Brandão, B. C., Lepkison, H., Souza, J. A. M. F., Corrêa, J. F. S., **Genetic Algorithms applied for Pattern Generation for Downhole Dynamometer Cards**, 2004.

Sovierzoski, M. A., Argoud, F. I. M., Azevedo, F. M., **Avaliação do Classificador Neural Binário com Análise ROC**, 21º Congresso Brasileiro de Engenharia Biomédica - CBEB, 2008, Salvador. Anais do 21º Congresso Brasileiro de Engenharia Biomédica-CBEB, 2008. p. 1-4.

Takács, G., **Sucker-rod Pumping Manual**, PennWell Books, 2002.

Thomas, J. E. (organizador) *et al.*, **Fundamentos da Engenharia de Petróleo**, 2ª Edição, Editora Interciência, Petrobras, Rio de Janeiro, 2001.

Witten, Ian H., Frank, Eibe. **Data Mining: Practical Machine Learning Tools and Techniques**, Second Edition, Elsevier Inc., 2005.

Xu, P., Xu, S., Yin, H. **Application of self-organizing competitive neural network in fault diagnosis of suck rod pumping system**, Journal of Petroleum Science and Engineering, Volume 58, Issues 1-2, August 2007, Pages 43-48

# Apêndice A

Neste apêndice, são apresentados os resultados dos testes realizados em detalhes.

## A. Descrição detalhada dos testes realizados

Todos os testes foram realizados utilizando-se um processador AMD Athlon 64 de 2.21GHz, 2 GB de memória RAM e o Windows XP como sistema operacional, exceto o teste A.1.1, que foi executado em uma máquina com dois processadores AMD Opteron 248 de 2.19 GHz cada, 16 GB de RAM e Windows XPx64.

Para realizar a implementação, o treinamento e a simulação das redes neurais apresentadas neste trabalho, utilizou-se o toolbox de redes neurais Neural Network Toolbox<sup>13</sup> do Matlab<sup>14</sup>.

Os resultados de cada teste realizado serão apresentados em uma tabela com as seguintes colunas:

- Unidades – Representa o número de neurônios na camada escondida;
- Melhor Época – A melhor época de um teste representa o ciclo a partir da qual a rede começa a “decorar” o conjunto de treinamento (quando ocorre o *overfitting*). Como são realizados 30 testes para cada quantidade de neurônios na camada escondida, a melhor época apresentada em cada linha será uma média da melhor época desses 30 testes.
- MSE Treino – Representa o erro de classificação da rede para o conjunto de treinamento. O erro é medido pela métrica MSE (*Mean Squared Error*), vista no capítulo 2.
- MSE Validação – Representa o erro de classificação da rede para o conjunto de validação.
- MSE Teste – Representa o erro de classificação da rede para o conjunto de teste.

---

<sup>13</sup> <http://www.mathworks.com/products/neuralnet/>

<sup>14</sup> <http://www.mathworks.com/>

## A.1 Redução de atributos inicial: eliminação das posições

A primeira redução de atributos realizada foi a eliminação das coordenadas de posição da carta dinamométrica. A seguir são apresentados os testes com a rede que utiliza como entrada 200 atributos de carga e posição, os testes com a rede que utiliza somente os atributos de carga e os testes com a rede que utiliza somente os atributos de posição.

### A.1.1 Rede utilizando 200 atributos (100 cargas e 100 posições)

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	11,4000	0,1344	0,1492	0,1544
2	16,3333	0,0756	0,1022	0,1088
3	18,4333	0,0225	0,0570	0,0668
4	16,9000	0,0138	0,0590	0,0629
5	29,5667	0,0144	0,0549	0,0608
6	13,6667	0,0149	0,0543	0,0560
7	20,1667	0,0094	0,0478	0,0587
8	46,3333	0,0130	0,0517	0,0531
9	15,3000	0,0151	0,0473	0,0546
10	20,4667	0,0099	0,0495	0,0531
11	81,7333	0,0091	0,0504	0,0565
12	47,4333	0,0124	0,0503	0,0526
13	12,1667	0,0124	0,0516	0,0572
14	12,3000	0,0102	0,0509	0,0506
15	12,3000	0,0076	0,0484	0,0521
16	11,1333	0,0080	0,0501	0,0502
17	26,6000	0,0073	0,0491	0,0520
18	11,1333	0,0056	0,0473	0,0471
19	10,7667	0,0086	0,0509	0,0527
20	14,4333	0,0064	0,0496	0,0502
25	9,5000	0,0046	0,0498	0,0491
30	16,3667	0,0036	0,0443	0,0401
40	10,5000	0,0033	0,0439	0,0382
50	48,6000	0,0027	0,0416	0,0378
Média	22,2306	0,0177	0,0563	0,0590
Desvio padrão	17,3452	0,0286	0,0228	0,0244
Mínimo	9,5000	0,0027	0,0416	0,0378
Máximo	81,7333	0,1344	0,1492	0,1544
Tempo	708.878,40s = 8d 4h 54min 38,40s			

**Tabela A.1.** Resultado da Rede com 200 atributos (100 cargas e 100 posições).

### A.1.2 Rede utilizando 100 atributos (100 cargas)

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	10,1333	0,1325	0,1475	0,1492
2	9,3333	0,0882	0,1115	0,1136
3	11,2000	0,0356	0,0761	0,0815
4	45,3667	0,0226	0,0666	0,0772
5	9,8000	0,0173	0,0633	0,0687
6	11,2333	0,0219	0,0688	0,0716
7	12,3333	0,0175	0,0661	0,0670
8	11,4333	0,0205	0,0572	0,0635
9	13,3333	0,0182	0,0587	0,0594
10	62,1000	0,0199	0,0613	0,0637
11	11,5000	0,0181	0,0553	0,0559
12	12,0667	0,0154	0,0543	0,0626
13	10,9333	0,0154	0,0553	0,0644
14	14,5667	0,0152	0,0526	0,0553
15	14,0333	0,0172	0,0576	0,0597
16	11,5333	0,0151	0,0562	0,0563
17	10,1333	0,0163	0,0586	0,0593
18	10,0333	0,0143	0,0568	0,0592
19	9,0667	0,0174	0,0572	0,0571
20	10,2000	0,0157	0,0540	0,0574
25	11,0333	0,0129	0,0583	0,0532
30	10,5667	0,0110	0,0561	0,0611
40	10,0000	0,0070	0,0616	0,0616
50	9,1000	0,0069	0,0613	0,0639
Média	14,6264	0,0247	0,0655	0,0684
Desvio padrão	12,3774	0,0277	0,0211	0,0212
Mínimo	9,0667	0,0069	0,0526	0,0532
Máximo	62,1000	0,1325	0,1475	0,1492
Tempo	81486,36s = 22h 38min 6,36s			

Tabela A.2. Resultado da Rede com 100 atributos (somente cargas).

### A.1.3 Rede utilizando 100 atributos (100 posições)

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	8,9000	0,1374	0,1541	0,1607
2	10,0333	0,0800	0,1053	0,1165
3	10,1333	0,0333	0,0774	0,0832
4	11,4333	0,0299	0,0811	0,0878
5	15,4667	0,0267	0,0715	0,0805
6	16,9667	0,0153	0,0622	0,0730
7	11,7333	0,0207	0,0604	0,0676
8	46,7000	0,0224	0,0647	0,0796
9	11,1667	0,0195	0,0619	0,0686
10	12,5667	0,0226	0,0596	0,0671
11	13,2000	0,0160	0,0594	0,0635

12	11,9000	0,0168	0,0524	0,0557
13	13,3667	0,0164	0,0551	0,0566
14	19,5667	0,0148	0,0510	0,0546
15	10,5667	0,0183	0,0551	0,0579
16	12,0333	0,0166	0,0556	0,0585
17	12,0000	0,0162	0,0603	0,0617
18	11,6333	0,0158	0,0606	0,0625
19	10,7000	0,0145	0,0578	0,0594
20	10,3667	0,0182	0,0581	0,0605
25	15,3667	0,0118	0,0550	0,0583
30	10,4667	0,0140	0,0581	0,0590
40	12,8667	0,0101	0,0655	0,0611
50	14,0000	0,0101	0,0689	0,0629
Média	13,8806	0,0257	0,0671	0,0715
Desvio padrão	7,4010	0,0275	0,0218	0,0235
Mínimo	8,9000	0,0101	0,0510	0,0546
Máximo	46,7000	0,1374	0,1541	0,1607
Tempo	95724,73s = 1d 2h 35min 24,73s			

**Tabela A.3.** Resultado da Rede com 100 atributos (somente posições).

#### A.1.4 Análise da redução inicial de atributos

Como se esperava, o MSE não aumentou significativamente com a retirada das coordenadas de posições. O tempo de treinamento se reduziu bruscamente. É importante lembrar que o algoritmo de treinamento utilizado nesses testes foi o Levenberg-Marquardt. Conforme explicado no capítulo 3, ele é indicado para redes pequenas por necessitar de grande quantidade de memória disponível. Por essa razão, para rodar os testes com a rede de 200 entradas, foi necessário um computador com maior poder de processamento e memória disponível do que para rodar as demais redes, como descrito no início deste apêndice.

Pelo mesmo motivo, a diferença entre o tempo de treinamento da rede com 200 entradas e o tempo de treinamento das redes com 100 entradas foi tão acentuada. Provavelmente, se outro algoritmo fosse utilizado, a diferença entre os tempos seria menor que a obtida com o Levenberg-Marquardt, mas continuaria grande o suficiente para justificar a redução de atributos.

## A.2 Continuação da redução de atributos

### A.2.1 Rede utilizando 50 cargas selecionadas pelo Método Alternado

Os 50 atributos de carga foram escolhidos alternadamente, isto é, escolheu-se 1 atributo a cada 2.

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	10,7000	0,1305	0,1407	0,1446
2	11,1667	0,0830	0,1042	0,1091
3	11,2667	0,0357	0,0712	0,0766
4	10,9000	0,0216	0,0595	0,0649
5	10,7000	0,0191	0,0553	0,0701
6	12,8000	0,0215	0,0568	0,0641
7	12,6000	0,0195	0,0589	0,0630
8	12,6667	0,0188	0,0551	0,0588
9	13,2667	0,0184	0,0529	0,0565
10	13,2333	0,0184	0,0539	0,0574
11	11,5333	0,0185	0,0543	0,0581
12	13,4000	0,0148	0,0544	0,0559
13	12,1333	0,0184	0,0506	0,0565
14	11,2333	0,0197	0,0545	0,0599
15	12,2333	0,0139	0,0530	0,0594
16	10,1000	0,0199	0,0546	0,0576
17	10,8667	0,0176	0,0514	0,0547
18	10,1000	0,0202	0,0532	0,0577
19	10,6667	0,0162	0,0523	0,0552
20	11,0667	0,0168	0,0558	0,0651
25	10,8333	0,0159	0,0554	0,0557
30	10,9333	0,0145	0,0576	0,0555
40	8,7667	0,0112	0,0582	0,0620
50	9,2333	0,0092	0,0649	0,0568
Média	11,3500	0,0256	0,0616	0,0656
Desvio padrão	1,2310	0,0264	0,0200	0,0202
Mínimo	8,7667	0,0092	0,0506	0,0547
Máximo	13,4000	0,1305	0,1407	0,1446
Tempo	17612,03s = 4h 53min 32,03s			

Tabela A.4. Resultado da Rede com 50 atributos - Método Alternado.

### A.2.2 Rede utilizando 50 cargas selecionadas pelo Método DesvPad

Foram escolhidos as 50 cargas que possuíam valores mais alto de desvio padrão entre as 100 cargas.

Unidades	Melhor	MSE Treino	MSE	MSE Teste
----------	--------	------------	-----	-----------

	Época		Validação	
1	9,1000	0,1336	0,1455	0,1462
2	8,6667	0,0842	0,1055	0,1053
3	11,0333	0,0316	0,0684	0,0709
4	10,8333	0,0283	0,0650	0,2971
5	9,7667	0,0270	0,0634	0,0652
6	12,7333	0,0212	0,0568	0,0674
7	12,2000	0,0218	0,0535	0,0651
8	14,0000	0,0229	0,0558	0,0640
9	11,7000	0,0200	0,0538	0,0585
10	13,6000	0,0192	0,0507	0,0583
11	12,0333	0,0224	0,0543	0,0598
12	11,3333	0,0218	0,0501	0,0529
13	12,4333	0,0178	0,0510	0,0593
14	11,7000	0,0217	0,0501	0,0537
15	12,5667	0,0188	0,0501	0,0547
16	10,0333	0,0207	0,0495	0,0551
17	12,1667	0,0165	0,0552	0,0579
18	10,0667	0,0188	0,0535	0,0557
19	11,6667	0,0192	0,0524	0,0539
20	11,3333	0,0165	0,0515	0,0548
25	10,8667	0,0175	0,0517	0,0539
30	12,0000	0,0130	0,0483	0,0511
40	10,7000	0,0137	0,0493	0,0524
50	10,9667	0,0099	0,0508	0,0558
Média	11,3958	0,0274	0,0598	0,0737
Desvio padrão	1,2946	0,0266	0,0217	0,0519
Mínimo	8,6667	0,0099	0,0483	0,0511
Máximo	14,0000	0,1336	0,1455	0,2971
Tempo	18645,79s = 5h 10min 45,79s			

Tabela A.5. Resultado da Rede com 50 atributos - Método DesvPad.

### A.2.3 Rede utilizando 30 cargas selecionadas pelo Método Alternado

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	13,8000	0,1317	0,1396	0,1405
2	10,8667	0,0798	0,0965	0,1002
3	10,6000	0,0360	0,0668	0,0657
4	11,6333	0,0230	0,0549	0,0676
5	12,0333	0,0192	0,0549	0,0649
6	14,7000	0,0200	0,0487	0,0508
7	12,6000	0,0234	0,0523	0,0579
8	12,8000	0,0166	0,0524	0,0599
9	12,0667	0,0191	0,0510	0,0554
10	13,0667	0,0183	0,0500	0,0596
11	12,8000	0,0191	0,0500	0,0507
12	10,3000	0,0195	0,0540	0,0561
13	11,9000	0,0184	0,0527	0,0541
14	11,0667	0,0183	0,0526	0,0558
15	11,6333	0,0168	0,0515	0,0673
16	12,0333	0,0167	0,0511	0,0564

17	12,5000	0,0193	0,0527	0,0584
18	10,3667	0,0191	0,0527	0,0608
19	11,3000	0,0183	0,0523	0,0563
20	11,8333	0,0163	0,0479	0,0576
25	12,0667	0,0147	0,0511	0,0560
30	10,5000	0,0174	0,0540	0,0576
40	10,1000	0,0152	0,0641	0,0649
50	10,0000	0,0090	0,0602	0,0569
Média	11,7736	0,0261	0,0589	0,0638
Desvio padrão	1,1918	0,0261	0,0198	0,0190
Mínimo	10,0000	0,0090	0,0479	0,0507
Máximo	14,7000	0,1317	0,1396	0,1405
Tempo	7149,33s = 1h 59min 9,33s			

Tabela A.6. Resultado da Rede com 30 atributos - Método Alternado.

#### A.2.4 Rede utilizando 30 cargas selecionadas pelo Método DesvPad

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	11,8333	0,1350	0,1449	0,1475
2	12,8667	0,0886	0,1075	0,1176
3	11,8000	0,0424	0,0736	0,0829
4	11,9333	0,0400	0,0689	0,0796
5	12,3000	0,0315	0,0616	0,0732
6	12,3667	0,0340	0,0624	0,0707
7	13,5333	0,0303	0,0583	0,0665
8	13,5667	0,0270	0,0596	0,0651
9	14,2667	0,0277	0,0584	0,0681
10	12,8333	0,0329	0,0609	0,0674
11	13,3000	0,0296	0,0570	0,0647
12	13,3000	0,0274	0,0564	0,0625
13	15,5000	0,0262	0,0549	0,0612
14	12,2667	0,0288	0,0570	0,0625
15	12,6000	0,0271	0,0546	0,0579
16	13,2333	0,0250	0,0536	0,0572
17	12,9667	0,0263	0,0553	0,0589
18	11,4000	0,0275	0,0557	0,0603
19	12,8000	0,0253	0,0553	0,0561
20	12,7667	0,0235	0,0538	0,0576
25	13,3667	0,0185	0,0564	0,0600
30	11,0667	0,0221	0,0564	0,0602
40	12,0667	0,0183	0,0569	0,0596
50	11,6000	0,0172	0,0591	0,0642
Média	12,7306	0,0347	0,0641	0,0701
Desvio padrão	0,9754	0,0254	0,0204	0,0209
Mínimo	11,0667	0,0172	0,0536	0,0561
Máximo	15,5000	0,1350	0,1449	0,1475
Tempo	8907,69s = 2h 28min 27,69s			

Tabela A.7. Resultado da Rede com 30 atributos - Método DesvPad.

## A.2.5 Rede utilizando 20 cargas selecionadas pelo Método Alternado

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	16,5667	0,1307	0,1371	0,1364
2	13,3667	0,0810	0,0953	0,0927
3	14,1000	0,0335	0,0580	0,0608
4	11,3333	0,0270	0,0518	0,0474
5	11,3333	0,0298	0,0535	0,0543
6	13,5000	0,0289	0,0521	0,0556
7	11,2667	0,0273	0,0536	0,0544
8	11,8000	0,0249	0,0530	0,0514
9	12,0667	0,0245	0,0522	0,0573
10	12,3000	0,0245	0,0536	0,0588
11	11,4333	0,0245	0,0514	0,0490
12	12,2667	0,0218	0,0521	0,0574
13	12,4333	0,0239	0,0512	0,0486
14	12,1333	0,0219	0,0509	0,0557
15	12,0333	0,0202	0,0530	0,0542
16	11,8333	0,0208	0,0521	0,0503
17	11,0000	0,0203	0,0518	0,0497
18	11,3333	0,0214	0,0514	0,0515
19	12,6333	0,0211	0,0522	0,0499
20	11,2333	0,0215	0,0561	0,0561
25	12,1000	0,0168	0,0508	0,0561
30	9,3333	0,0200	0,0544	0,0537
40	10,3667	0,0164	0,0526	0,0592
50	9,7000	0,0146	0,0580	0,0606
Média	11,9778	0,0299	0,0583	0,0592
Desvio padrão	1,4607	0,0249	0,0190	0,0186
Mínimo	9,3333	0,0146	0,0508	0,0474
Máximo	16,5667	0,1307	0,1371	0,1364
Tempo	3750,79s = 1h 2min 30,79s			

Tabela A.8. Resultado da Rede com 20 atributos - Método Alternado.

## A.2.6 Rede utilizando 20 cargas selecionadas pelo Método DesvPad

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	13,1667	0,1416	0,1466	0,1498
2	11,2333	0,1106	0,1253	0,1308
3	17,1667	0,0771	0,1029	0,1107
4	17,2000	0,0661	0,0928	0,1160
5	15,9000	0,0666	0,0934	0,1020
6	14,7000	0,0655	0,0927	0,1012
7	13,5667	0,0640	0,0901	0,1008
8	14,7333	0,0581	0,0869	0,0993
9	14,0667	0,0589	0,0886	0,0980
10	13,8667	0,0571	0,0854	0,0987
11	14,2000	0,0545	0,0850	0,0962
12	14,3333	0,0498	0,0811	0,0965

13	14,4667	0,0494	0,0825	0,0938
14	14,5000	0,0507	0,0812	0,0989
15	13,9000	0,0492	0,0828	0,0988
16	14,2667	0,0468	0,0812	0,0961
17	13,2667	0,0531	0,0821	0,0933
18	13,4333	0,0471	0,0803	0,0953
19	14,8000	0,0474	0,0772	0,0906
20	14,6333	0,0448	0,0782	0,0922
25	13,8000	0,0441	0,0777	0,0918
30	16,0000	0,0402	0,0771	0,0906
40	15,4667	0,0383	0,0748	0,0880
50	14,2000	0,0387	0,0741	0,0913
Média	14,4528	0,0592	0,0883	0,1009
Desvio padrão	1,2748	0,0232	0,0165	0,0139
Mínimo	11,2333	0,0383	0,0741	0,0880
Máximo	17,2000	0,1416	0,1466	0,1498
Tempo	5774,10s = 1h 36min 14,10s			

Tabela A.9. Resultado da Rede com 20 atributos - Método DesvPad.

### A.2.7 Rede utilizando 10 cargas selecionadas pelo Método Alternado

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	11,9333	0,1393	0,1433	0,1472
2	17,6667	0,0992	0,1099	0,1123
3	12,3000	0,0524	0,0737	0,0756
4	12,4000	0,0511	0,0728	0,0724
5	14,8000	0,0453	0,0686	0,0686
6	13,4000	0,0430	0,0645	0,0602
7	14,3333	0,0399	0,0645	0,0646
8	14,9667	0,0374	0,0648	0,0631
9	13,8333	0,0349	0,0631	0,0671
10	13,4000	0,0353	0,0649	0,0654
11	13,4667	0,0349	0,0622	0,0649
12	12,0333	0,0348	0,0607	0,0643
13	12,4000	0,0352	0,0590	0,0622
14	11,8000	0,0347	0,0609	0,0615
15	13,6667	0,0310	0,0600	0,0647
16	10,9000	0,0324	0,0606	0,0641
17	11,8667	0,0316	0,0604	0,0613
18	11,2333	0,0323	0,0598	0,0629
19	10,8000	0,0320	0,0608	0,0620
20	11,0000	0,0311	0,0589	0,0587
25	11,7333	0,0281	0,0609	0,0652
30	10,5000	0,0283	0,0566	0,0591
40	9,5000	0,0280	0,0648	0,0620
50	10,4667	0,0215	0,0669	0,0637
Média	12,5167	0,0422	0,0684	0,0697
Desvio padrão	1,8134	0,0254	0,0190	0,0196
Mínimo	9,5000	0,0215	0,0566	0,0587
Máximo	17,6667	0,1393	0,1433	0,1472

Tempo	2429,30s = 40min 29,30s
-------	-------------------------

**Tabela A.10.** Resultado da Rede com 10 atributos - Método Alternado.

### A.2.8 Rede utilizando 10 cargas selecionadas pelo Método DesvPad

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	16,9333	0,1449	0,1495	0,1534
2	18,4000	0,1249	0,1357	0,1417
3	14,8667	0,1129	0,1265	0,1353
4	15,5000	0,1046	0,1185	0,1291
5	14,9667	0,1003	0,1163	0,1318
6	17,0667	0,0936	0,1096	0,1258
7	17,7000	0,0911	0,1074	0,1261
8	15,2000	0,0907	0,1099	0,1231
9	16,7667	0,0875	0,1077	0,1296
10	15,6667	0,0865	0,1079	0,1226
11	17,3000	0,0836	0,1068	0,1169
12	17,8333	0,0805	0,1042	0,1284
13	17,4333	0,0815	0,1037	0,1180
14	17,4333	0,0790	0,1033	0,1216
15	16,9000	0,0789	0,1030	0,1164
16	15,2000	0,0817	0,1045	0,1193
17	18,1000	0,0777	0,1041	0,1175
18	17,1333	0,0775	0,1045	0,1218
19	17,1333	0,0784	0,1044	0,1191
20	17,7333	0,0771	0,1032	0,1184
25	16,9667	0,0729	0,1014	0,1140
30	15,1333	0,0752	0,1031	0,1199
40	15,6000	0,0713	0,1023	0,1150
50	13,6000	0,0737	0,1032	0,1152
Média	16,5236	0,0886	0,1100	0,1242
Desvio padrão	1,2549	0,0178	0,0118	0,0093
Mínimo	13,6000	0,0713	0,1014	0,1140
Máximo	18,4000	0,1449	0,1495	0,1534
Tempo	2900,48s = 48min 20,48s			

**Tabela A.11.** Resultado da Rede com 10 atributos - Método DesvPad

### A.2.9 Análise da redução de atributos

Nas tabelas abaixo, tem-se um resumo dos testes realizados antes e depois da redução de atributos:

- Antes de utilizar os métodos Alternado e DesvPad para redução de atributos:

Número Atributos	MSE Médio	Desvio Padrão	Tempo (s)
100	0,0684	0,0212	81486,36

**Tabela A.12.** Resultado dos Testes com 100 atributos de carga.

- Resultado da redução de atributos pelos métodos Alternado e DesvPad:

Número Atributos	Método Alternado			Método DesvPad		
	MSE Médio	Desvio Padrão	Tempo (s)	MSE Médio	Desvio Padrão	Tempo (s)
50	0,0656	0,0202	17612,03	0,0737	0,0519	18645,79
30	0,0638	0,0190	7149,33	0,0701	0,0209	8907,69
20	0,0592	0,0186	3750,79	0,1009	0,0139	5774,10
10	0,0697	0,0196	2429,30	0,1242	0,0093	2900,48

Tabela A.13. Resumo das Reduções de Atributos.

### A.3 Uma rede com N saídas x N redes com uma saída

Os próximos testes avaliam possibilidade de se ter uma rede especialista para cada situação operacional de um sistema de bombeio mecânico ao invés de se ter uma única rede com várias saídas, uma para cada falha.

Abaixo, serão apresentados os resultados independentes de quatro redes com 20 entradas: uma especializada em identificar se uma carta representa uma situação Normal, a outra em identificar se a carta representa uma Pancada de Fluido, uma terceira especialista em cartas com Interferência de Gás e, finalmente, uma especializada em reconhecer cartas diferentes das demais (Outra Classificação).

#### A.3.1 Rede Neural Especialista em CDF Normal

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	9,9667	0,0235	0,0488	0,0799
2	9,3333	0,0227	0,0481	0,0660
3	42,3000	0,0163	0,0446	0,0695
4	42,4667	0,0135	0,0325	0,0521
5	44,2333	0,0108	0,0359	0,0479
6	96,7333	0,0066	0,0323	0,0513
7	65,0333	0,0149	0,0347	0,0632
8	80,6667	0,0068	0,0328	0,0607
9	11,0000	0,0084	0,0335	0,0519
10	45,7667	0,0061	0,0285	0,0859
11	86,6667	0,0059	0,0259	0,0563
12	43,1333	0,0097	0,0398	0,0635
13	10,3000	0,0076	0,0323	0,0673
14	10,8667	0,0097	0,0402	0,0705
15	12,0667	0,0078	0,0390	0,0712

16	9,7000	0,0115	0,0346	0,0639
17	9,5667	0,0055	0,0417	0,0651
18	9,2000	0,0084	0,0425	0,0774
19	12,9667	0,0082	0,0388	0,0964
20	8,3667	0,0085	0,0450	0,0642
25	7,1000	0,0115	0,0487	0,0665
30	7,4000	0,0076	0,0496	0,0810
40	6,3000	0,0086	0,0586	0,0983
50	5,8333	0,0094	0,0648	0,0915
Média	28,6236	0,0104	0,0406	0,0692
Desvio padrão	28,5979	0,0048	0,0093	0,0139
Mínimo	5,8333	0,0055	0,0259	0,0479
Máximo	96,7333	0,0235	0,0648	0,0983
Tempo	1307,57s = 21min 47,57s			

**Tabela A.14.** Resultado da rede neural especializada em cartas Normais

### A.3.2 Rede Neural Especialista em CDF com Pancada de Fluido

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	5,1333	0,0848	0,1221	0,1246
2	6,5000	0,0707	0,1127	0,1229
3	9,6667	0,0686	0,1123	0,1137
4	7,2333	0,0666	0,1057	0,1211
5	6,4667	0,0736	0,1059	0,1222
6	7,6333	0,0546	0,1081	0,1268
7	6,6333	0,0638	0,1048	0,1234
8	5,7333	0,0629	0,1166	0,1494
9	6,0333	0,0602	0,1053	0,1185
10	7,7667	0,0429	0,0996	0,1276
11	6,3667	0,0525	0,1128	0,1430
12	7,9667	0,0434	0,1060	0,1484
13	7,2667	0,0528	0,1136	0,1455
14	6,8000	0,0599	0,1051	0,1429
15	5,9333	0,0577	0,1183	0,1288
16	5,8333	0,0565	0,1060	0,1292
17	5,8000	0,0469	0,1135	0,1455
18	7,2667	0,0475	0,1197	0,1384
19	4,7333	0,0599	0,1083	0,1324
20	5,5333	0,0393	0,1139	0,1449
25	4,5667	0,0619	0,1218	0,1485
30	6,0667	0,0370	0,1309	0,1789
40	5,2667	0,0330	0,1345	0,1686
50	5,1000	0,0298	0,1452	0,2051
Média	6,3875	0,0553	0,1143	0,1396
Desvio padrão	1,1795	0,0135	0,0107	0,0210
Mínimo	4,5667	0,0298	0,0996	0,1137
Máximo	9,6667	0,0848	0,1452	0,2051
Tempo	1030,03s = 17min 10,03s			

**Tabela A.15.** Resultado da rede neural especializada em cartas com Pancada de Fluido.

### A.3.3 Rede Neural Especialista em CDF com Interferência de Gás

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	5,4000	0,0558	0,0739	0,0772
2	45,5667	0,0420	0,0678	0,2063
3	7,0000	0,0336	0,0639	0,0864
4	9,0667	0,0312	0,0669	0,0669
5	10,2333	0,0323	0,0660	0,0746
6	8,5000	0,0339	0,0596	0,0739
7	8,6333	0,0347	0,0694	0,0725
8	6,9333	0,0367	0,0620	0,0603
9	8,6000	0,0300	0,0629	0,0622
10	8,7333	0,0365	0,0663	0,0750
11	7,8333	0,0281	0,0663	0,0669
12	7,8333	0,0258	0,0644	0,0630
13	7,0667	0,0247	0,0639	0,0651
14	7,6000	0,0249	0,0648	0,0582
15	6,6667	0,0248	0,0616	0,0675
16	6,3000	0,0259	0,0641	0,0682
17	7,1667	0,0258	0,0702	0,0749
18	6,8667	0,0227	0,0658	0,0605
19	7,2000	0,0208	0,0638	0,0641
20	5,9333	0,0247	0,0674	0,0610
25	7,3000	0,0191	0,0775	0,0829
30	6,4000	0,0227	0,1011	0,1183
40	5,8333	0,0177	0,0931	0,0950
50	6,2000	0,0344	0,1265	0,1158
Média	8,9528	0,0295	0,0712	0,0799
Desvio padrão	7,8841	0,0083	0,0152	0,0313
Mínimo	5,4000	0,0177	0,0596	0,0582
Máximo	45,5667	0,0558	0,1265	0,2063
Tempo	1041,34s = 17min 21,34s			

Tabela A.15. Resultado da rede neural especializada em cartas com Interferência de Gás.

### A.3.4 Rede Neural Especialista em CDF com Outra Classificação

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	43,4667	0,0185	0,0299	0,0423
2	84,7667	0,0057	0,0195	0,0402
3	56,2000	0,0103	0,0218	0,0276
4	86,0333	0,0113	0,0233	0,0317
5	144,7333	0,0068	0,0205	0,0273
6	249,5667	0,0099	0,0197	0,0303
7	163,7667	0,0132	0,0230	0,0343
8	119,1000	0,0079	0,0254	0,0281
9	59,0000	0,0084	0,0229	0,0281
10	88,5667	0,0088	0,0217	0,0303
11	117,2000	0,0042	0,0202	0,0229
12	95,2000	0,0087	0,0259	0,0281

13	13,6333	0,0083	0,0280	0,0275
14	46,3000	0,0084	0,0261	0,0288
15	47,6667	0,0117	0,0314	0,0302
16	11,4333	0,0090	0,0314	0,0355
17	11,0000	0,0103	0,0295	0,0338
18	45,1333	0,0068	0,0309	0,0386
19	11,9333	0,0119	0,0357	0,0346
20	10,6667	0,0078	0,0334	0,0363
25	10,3000	0,0093	0,0351	0,0419
30	10,9333	0,0100	0,0439	0,0483
40	8,8667	0,0102	0,0433	0,0391
50	8,3667	0,0108	0,0447	0,0431
Média	64,3264	0,0095	0,0286	0,0337
Desvio padrão	61,2216	0,0028	0,0077	0,0064
Mínimo	8,3667	0,0042	0,0195	0,0229
Máximo	249,5667	0,0185	0,0447	0,0483
Tempo	3021,76s = 50min 21,76s			

Tabela A.16. Resultado da rede neural especializada em cartas com outra classificação.

## A.4 Algoritmo de Treinamento: Levenberg-Marquardt x Gradiente Descendente

Os testes anteriores foram realizados com o algoritmo Levenberg-Marquardt (LM)<sup>15</sup>. Na tabela A.17, será mostrado o resultado do teste com a rede de 20 atributos (escolhidos segundo o Método Alternado) com o algoritmo Gradiente Descendente (GD)<sup>16</sup> para verificar se há alteração significativa no MSE obtido no conjunto de teste.

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	161,3000	0,1393	0,1433	0,1449
2	152,4667	0,1203	0,1256	0,1251
3	165,7000	0,0849	0,0939	0,0938
4	163,4333	0,0680	0,0792	0,0802
5	163,8667	0,0598	0,0721	0,0729
6	159,5000	0,0641	0,0754	0,0749
7	159,8333	0,0608	0,0722	0,0736
8	161,6000	0,0583	0,0697	0,0714
9	160,6667	0,0578	0,0693	0,0692
10	158,9000	0,0582	0,0682	0,0708
11	158,2667	0,0575	0,0683	0,0705
12	154,2333	0,0584	0,0705	0,0714

<sup>15</sup> No Neural Network Toolbox do Matlab, o método de Levenberg-Marquardt é o algoritmo padrão (*default*) de treinamento e é representado pelo termo “**trainlm**”

<sup>16</sup> Para permitir que a rede seja treinada com o algoritmo Gradiente Descendente no Neural Network Toolbox do Matlab, deve-se alterar a rede antes de treiná-la da seguinte forma: **net.trainFcn = 'traingdx'**.

13	157,3667	0,0577	0,0692	0,0706
14	156,0000	0,0580	0,0683	0,0698
15	154,1000	0,0585	0,0696	0,0700
16	155,5667	0,0574	0,0688	0,0707
17	153,6000	0,0572	0,0692	0,0710
18	152,4333	0,0582	0,0685	0,0700
19	151,0000	0,0588	0,0699	0,0709
20	150,9000	0,0568	0,0683	0,0693
25	145,7667	0,0580	0,0696	0,0717
30	139,0333	0,0632	0,0747	0,0751
40	129,1333	0,0839	0,1018	0,1005
50	98,7000	0,1868	0,2035	0,2039
Média	152,6403	0,0726	0,0837	0,0847
Desvio padrão	14,0658	0,0320	0,0320	0,0317
Mínimo	98,7000	0,0568	0,0682	0,0692
Máximo	165,7000	0,1868	0,2035	0,2039
Tempo	1590,80s = 26min 30,80s			

**Tabela A.17.** Rede treinada com algoritmo Gradiente Descendente (6 ciclos).

A melhor época nos testes com o GD ocorre, em média, por volta da 152<sup>a</sup> época, com um desvio padrão de 14 épocas aproximadamente. Já com o LM, em média, a melhor época ocorre na 12<sup>a</sup> época, com desvio padrão também aproximado de 1,5 de épocas (os resultados do LM estão no item A.2.5).

Embora seja necessário um número menor de ciclos para convergir com o LM, percebeu-se, nos testes realizados, que o tempo de cada ciclo do LM é superior ao tempo de um ciclo do GD, já que o tempo total de duração dos testes com o LM foi o dobro do tempo com o GD. Talvez o tempo maior do LM se deva ao fato dele necessitar de uma grande quantia de memória disponível. A razão entre o tempo do LM e o tempo do GD aumenta à medida em que se aumenta a rede. Por isso, o LM é indicado para redes pequenas.

Comparando-se os erros médios obtidos com os dois algoritmos, percebe-se que o erro com o GD foi 43% maior do que o obtido com o LM. Para verificar se o erro com o GD pode ser reduzido se a configuração de treinamento for alterada, aumentou-se o número de ciclos – de 6 para 20 ciclos – em que se verifica o erro do conjunto de validação antes de realizar a parada antecipada. No toolbox de redes neurais do Matlab, essa operação equivale a alterar o **net.trainParam.max\_fail**. A tabela A.18 mostra o resultado desse teste.

Unidades	Melhor Época	MSE Treino	MSE Validação	MSE Teste
1	204,6667	0,1340	0,1377	0,1415
2	308,2333	0,0963	0,1043	0,1061
3	541,1000	0,0571	0,0687	0,0688
4	465,9333	0,0510	0,0627	0,0627
5	466,0333	0,0462	0,0593	0,0604
6	595,5000	0,0448	0,0584	0,0572
7	565,6000	0,0442	0,0577	0,0568
8	595,0333	0,0412	0,0542	0,0548
9	650,4333	0,0425	0,0560	0,0564
10	607,7333	0,0421	0,0554	0,0569
11	705,6667	0,0407	0,0551	0,0540
12	559,7000	0,0434	0,0567	0,0583
13	651,6000	0,0411	0,0540	0,0561
14	648,3000	0,0427	0,0554	0,0562
15	675,2000	0,0398	0,0543	0,0545
16	711,1333	0,0399	0,0541	0,0547
17	722,9000	0,0400	0,0542	0,0547
18	645,5000	0,0413	0,0560	0,0563
19	687,2667	0,0417	0,0551	0,0575
20	670,9667	0,0414	0,0556	0,0569
25	602,4333	0,0428	0,0559	0,0564
30	654,4667	0,0419	0,0559	0,0568
40	665,0333	0,0423	0,0574	0,0568
50	681,2333	0,0417	0,0567	0,0578
Média	595,0694	0,0492	0,0621	0,0629
Desvio padrão	125,7106	0,0214	0,0191	0,0197
Mínimo	204,6667	0,0398	0,0540	0,0540
Máximo	722,9000	0,1340	0,1377	0,1415
Tempo	6260,55s = 1h 44min 20,55s			

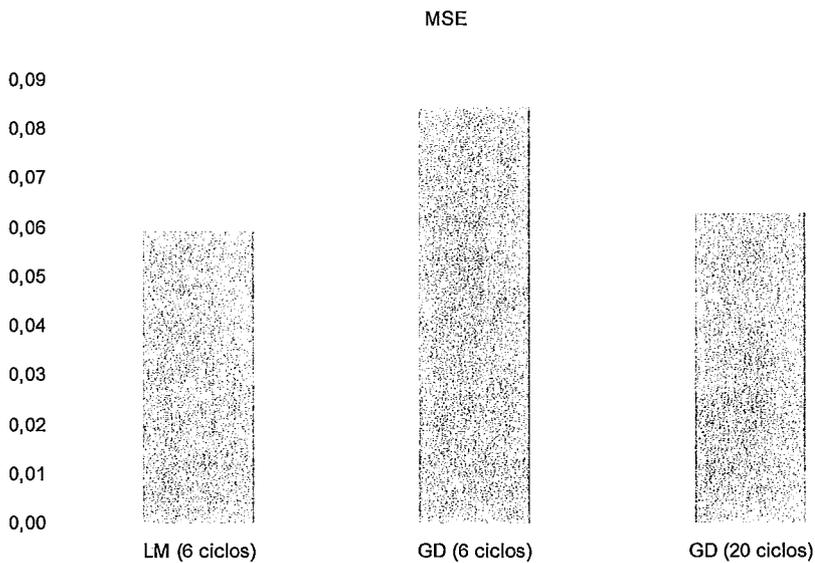
**Tabela A.18.** Rede treinada com algoritmo Gradiente Descendente (20 ciclos).

Com a nova configuração, o erro médio diminui mais de 25%. Isso mostra que, para se obter erros semelhantes com os dois algoritmos, o número de ciclos verificados na validação deve ser maior no treinamento do GD do que no do LM.

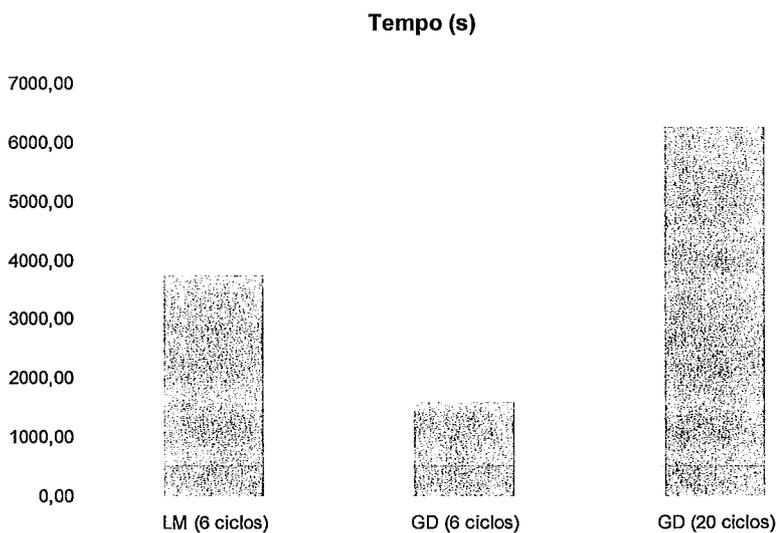
	MSE Validação	Desvio Padrão	Tempo (s)
LM (6 ciclos)	0,06	0,0190	3750,79
GD (6 ciclos)	0,08	0,0320	1590,81
GD (20 ciclos)	0,06	0,0191	6260,55

**Tabela A.19.** Gradiente Descendente x Levenberg-Marquardt.

Os gráficos A.1 e A.2 comparam o MSE e o tempo de treinamento, respectivamente, entre as diferentes aplicações dos algoritmos de treinamentos.



**Gráfico A.1.** MSE: Gradiente Descendente x Levenberg-Marquardt.



**Gráfico A.2.** Tempo de Treinamento: Gradiente Descendente x Levenberg-Marquardt.

## A.5 Redes Fuzzy ARTMAP

A tabela A.20 apresenta os resultados numéricos dos testes com o Fuzzy ARTMAP:

Taxa de Aprendizado	Acerto Geral	Acerto Treinamento	Acerto Teste
0,50	58,75	72,00	86,75
0,55	95,00	99,75	95,25

0,60	96,25	99,75	96,50
0,65	95,50	99,75	95,75
0,70	95,00	99,75	95,25
0,75	94,50	99,75	94,75
0,80	95,00	99,75	95,25
0,85	95,00	99,75	95,25
0,90	95,25	99,75	95,50
0,95	95,25	99,75	95,50
1,00	92,50	99,75	92,75

**Tabela A.20.** Acerto da Rede Fuzzy ARTMAP em função da Taxa de Aprendizado.

## A.6 Mapas de Kohonen

Esta seção descreve os testes realizados com mapas de Kohonen. Segue a explicação do cabeçalho das tabelas que contêm os resultados dos testes:

- **Mapa** – Número de unidades (neurônios) do mapa;
- **Ciclos** – Número de ciclos utilizados nos testes. Será representado por  $[C_1 C_2]$ , onde  $C_1$  é o número de ciclo da 1ª fase, onde se define a vizinhança; e  $C_2$  é o número de ciclo da 2ª fase, onde é feito o ajuste fino dos pesos;
- **Tempo** – Tempo, em segundos, de treinamento e avaliação do mapa.
- **qTrein** e **tTrein** – Erro médio de quantização e erro topológico, respectivamente, medidos com o mapa treinado de pesos e os dados do conjunto de treinamento;
- **qTeste** e **tTeste** – Erro médio de quantização e erro topológico, respectivamente, medidos com o mapa treinado de pesos e os dados do conjunto de teste;
- **a** – percentual de acertos;
- **e** – percentual de erros;
- **f** – percentual de fronteiras.

### Configuração inicial do mapa:

- Iniciação dos pesos: Linear
- Ciclos: [200 2000]
- Reticulado: Hexagonal
- Formato: Plano

Para a configuração descrita, foram realizados testes com mapas de diferentes tamanhos:

Mapa	Tempo	qTrein	tTrein	qTeste	tTeste	a	e	f
25x30	342,01	0,0849	0,0036	0,1359	0,0750	0,6428	0,0322	0,3250
20x25	164,79	0,1168	0,0321	0,1580	0,0583	0,7050	0,0450	0,2500
25x20	167,94	0,1163	0,0143	0,1499	0,0583	0,6692	0,0308	0,3000
20x15	77,29	0,1539	0,0107	0,1814	0,0333	0,7442	0,0725	0,1833
15x20	78,34	0,1553	0,0179	0,1850	0,0333	0,7517	0,0817	0,1667
10x20	46,97	0,1841	0,0036	0,2143	0,0250	0,7581	0,0836	0,1583
10x15	32,24	0,2072	0,0036	0,2321	0,0250	0,8457	0,1043	0,0500
10x12	27,80	0,2245	0,0107	0,2460	0,0000	0,8385	0,1198	0,0417
08x12	22,92	0,2468	0,0179	0,2730	0,0167	0,7690	0,1560	0,0750
07x10	19,70	0,2717	0,0107	0,3012	0,0250	0,7995	0,1755	0,0250

O melhor resultado – percentual de acertos igual a 84,57% – foi obtido com o mapa 10x15. Por isso, os demais testes serão realizados em cima desse mapa.

#### Alteração do número de ciclos de treinamento:

Ciclos	Tempo	qTrein	tTrein	qTeste	tTeste	a	e	f
[100 1000]	18,85	0,2063	0,0000	0,2366	0,0333	0,8083	0,1417	0,0500
[200 2000]	32,24	0,2072	0,0036	0,2321	0,0250	0,8457	0,1043	0,0500
[300 3000]	55,02	0,2073	0,0036	0,2321	0,0333	0,8429	0,1071	0,0500

#### Alteração do reticulado do mapa:

Reticulado	Tempo	qTrein	tTrein	qTeste	tTeste	a	e	f
Hexagonal	32,24	0,2072	0,0036	0,2321	0,0250	0,8457	0,1043	0,0500
Retangular	36,20	0,1994	0,1036	0,2227	0,0917	0,7884	0,1199	0,0917

#### Alteração do formato do mapa:

Forma	Tempo	qTrein	tTrein	qTeste	tTeste	a	e	f
Sheet	32,24	0,2072	0,0036	0,2321	0,0250	0,8457	0,1043	0,0500
Cyl	35,82	0,2102	0,0429	0,2412	0,0333	0,8242	0,1091	0,0667
Toroid	36,08	0,2277	0,0714	0,2680	0,1167	0,8152	0,1348	0,0500

#### Alteração da forma de iniciação dos pesos:

Iniciação dos Pesos	Medida	qTrein	tTrein	qTeste	tTeste	a	e	f
Lininit (linear)	Valor Único	0,2072	0,0036	0,2321	0,0250	0,8457	0,1043	0,0500
Randinit (randômica)	Média	0,2049	0,0111	0,2305	0,0217	0,8001	0,0915	0,1083
	Desvio Padrão	0,0028	0,0117	0,0012	0,0080	0,0134	0,0126	0,0171

A configuração que obteve o melhor resultado – 84,57% de acerto na classificação do conjunto de teste – foi:

- Mapa: 10 x 15
- Iniciação dos pesos: Linear
- Ciclos: [200 2000]
- Reticulado: Hexagonal
- Formato: Plano

# Apêndice B

Para avaliar se um mapa de Kohonen treinado pode ser utilizado para classificar novos padrões, utilizou-se as seguintes métricas: percentual de acertos ( $a$ ), percentual de erros ( $e$ ) e percentual de fronteira ( $f$ ). Este apêndice explica como cada uma dessas métricas é calculada.

## B. Métricas para avaliação da aplicação de um Mapa de Kohonen treinado como classificador

### B.1 Percentual de Acertos

O percentual de acertos ( $a$ ) é a razão da soma de todos os acertos pelo número de padrões do conjunto de teste:

$$a = \frac{\sum_{k=1}^m a(x_k)}{m}$$

Onde:

- $m$  é o tamanho do conjunto de teste;
- $x_k$  é o  $k$ -ésimo padrão do conjunto de teste ( $1 \leq k \leq m$ );
- $a(x_k)$  representa o acerto do padrão  $x_k$ .

O acerto de um padrão  $a(x_k)$  poderia ser igual a 1 quando o neurônio ativado por  $x_k$  tivesse sido ativado por outro padrão da sua classe; e, 0, caso contrário. Porém, sabe-se que um mesmo neurônio pode ser ativado por padrões de diferentes classes. Por isso, o acerto de um padrão  $a(x_k)$  será definido da seguinte forma:

$$a(x_k) = \frac{\# \text{ativações}_{\text{mesma classe}}}{\# \text{ativações}}$$

Onde:

- $\#ativações$  representa o número de vezes que o neurônio ativado por  $x_k$  fora acionado por padrões do conjunto de treinamento;
- $\#ativações_{mesma\ classe}$  representa o número de vezes que o neurônio ativado por  $x_k$  fora ativado por padrões do conjunto de treinamento que pertencem à classe de  $x_k$ .

## B.2 Percentual de Erros

O percentual de erros ( $e$ ) é calculado de forma semelhante ao percentual de acertos:

$$e = \frac{\sum_{k=1}^m e(x_k)}{m}, \text{ onde:}$$

$$e(x_k) = \frac{\#ativações_{classe\ diferente}}{\#ativações} = \frac{\#ativações - \#ativações_{mesma\ classe}}{\#ativações} = 1 - \frac{\#ativações_{mesma\ classe}}{\#ativações}$$

$$\therefore e(x_k) = 1 - a(x_k)$$

## B.3 Percentual de Fronteira

É importante lembrar que nem sempre todos os neurônios de um mapa são ativados durante o treinamento. Os neurônios que não são ativados, normalmente, fazem parte da fronteira que separa as classes (*clusters*). Portanto, quando uma amostra do conjunto de teste ativar um neurônio da fronteira, não se pode dizer que houve um erro ou um acerto. Poderia se utilizar a matriz U, que define a distância entre as unidades do mapa, para tentar identificar a classe que mais se aproxima daquele neurônio. Porém, por simplificação, quando uma amostra do conjunto de teste ativar um neurônio que não fora ativado antes, ficará registrado como uma ocorrência de fronteira. Assim, o percentual de fronteira ( $f$ ) pode ser definido como:

$$f = \frac{\sum_{k=1}^m f(x_k)}{m} \text{ onde } f(x_k) = \begin{cases} 1, & \text{neurônio} \in \text{fronteira} \\ 0, & \text{neurônio} \notin \text{fronteira} \end{cases}$$

## B.4 Propriedades das Métricas

Seguem algumas propriedades das métricas apresentadas:

- I. Para todo o conjunto de teste:  $a + e + f = 1$
- II. Para cada padrão  $x_k$  do conjunto de teste:
  - a.  $a(x_k) + e(x_k) + f(x_k) = 1$
  - b.  $a(x_k) + e(x_k) = 1$  ou  $f(x_k) = 1$ 
    - i.  $f(x_k) = 1 \Rightarrow a(x_k) = e(x_k) = 0$
    - ii.  $f(x_k) = 0 \Rightarrow a(x_k) + e(x_k) = 1$
    - iii.  $a(x_k) > 0 \Rightarrow e(x_k) = 1 - a(x_k)$  e  $f(x_k) = 0$

## B.5 Exemplo de aplicação

Será apresentado um exemplo de aplicação das métricas descritas acima, baseado na tabela B.1. Considere que um mapa de Kohonen tenha sido treinado e, em seguida, solicitado o BMU (neurônio mais próximo) para cada padrão do conjunto de treinamento e do conjunto de teste a partir desse mapa.

A tabela B.1 mostra cada padrão do conjunto de treinamento e do conjunto de teste, a classe à que pertence e o neurônio que ele ativa (BMU). A Figura B.1 mostra a representação gráfica dessa tabela.

Conjunto de Treinamento		
Padrão	Classe	BMU
P1	A	2
P2	A	2
P3	B	2
P4	C	3
P5	C	2
P6	B	1

Conjunto de Teste		
Padrão	Classe	BMU
T1	A	2
T2	B	4

Tabela B.1. Exemplo de conjuntos de treinamento e teste.

Conjunto de Treinamento		Conjunto de Teste	
Neurônio 1 P6(B)	Neurônio 3 P4(C)	Neurônio 1	Neurônio 3
Neurônio 2 P1(A), P2(A), P3(B), P5(C)	Neurônio 4	Neurônio 2 T1(A)	T2(B)

**Figura B.1.** Mapa mostrando qual neurônio cada padrão ativou.

A seguir, será apresentado como cada padrão conjunto de teste é analisado pelo seu BMU e como é a avaliação de todo o conjunto de teste. Por fim, avalia-se se o mapa de Kohonen do exemplo seria um bom classificador para novos padrões.

### Avaliação do padrão T1:

Há quatro padrões do conjunto de treinamento que ativam o mesmo neurônio que T1: P1, P2, P3 e P5. Desses quatro, dois pertencem a mesma classe de T1 e dois pertencem a outras classes. Assim, o acerto de T1 é igual a  $a(T1) = 2/4 = 0.5$  e o erro (pela propriedade iii) é igual a  $e(T1) = 1 - 0.5 = 0.5$ . T1 não ativou um neurônio da fronteira, portanto  $f(T1) = 0$ .

### Avaliação do padrão T2:

O neurônio ativado por T2 (o neurônio 4) não foi ativado pelos padrões do conjunto de treinamento. Isso significa que o neurônio pertence à fronteira. Assim,  $f(T2) = 1$ . Pela propriedade i,  $a(T2) = 0$  e  $e(T2) = 0$ .

### Avaliação do conjunto de teste:

$$a = \frac{\sum_{k=1}^m a(x_k)}{m} = \frac{\sum_{k=1}^2 a(x_k)}{2} = \frac{a(T1) + a(T2)}{2} = \frac{0.5 + 0}{2} = 0.25$$

$$e = \frac{\sum_{k=1}^m e(x_k)}{m} = \frac{\sum_{k=1}^2 e(x_k)}{2} = \frac{e(T1) + e(T2)}{2} = \frac{0.5 + 0}{2} = 0.25$$

$$f = \frac{\sum_{k=1}^m f(x_k)}{m} = \frac{\sum_{k=1}^2 f(x_k)}{2} = \frac{f(T1) + f(T2)}{2} = \frac{0+1}{2} = 0.5 \text{ ou, pela propriedade I,}$$

$$f = 1 - a - e = 1 - 0,25 - 0,25 = 0.5$$

### **Avaliação da aplicação do Mapa de Kohonen do exemplo como classificador:**

Um processo de validação que retorne valores semelhantes ao do exemplo pode mostrar que o mapa não será um bom classificador, já que apenas 25% dos padrões de teste ativaram neurônios ativados por padrões que pertencem a mesma classe. Provavelmente, ao se tentar classificar novos padrões, o mapa não terá sucesso na maioria das tentativas, visto que, na validação, 50% dos padrões ativaram neurônios de fronteira e, 25%, neurônios correspondentes a outras classificações.

## Apêndice C

Este apêndice explica a interface do SOLAR e a interface do Módulo IFBM.

A Figura C.1 apresenta a visão geral da interface do SOLAR.

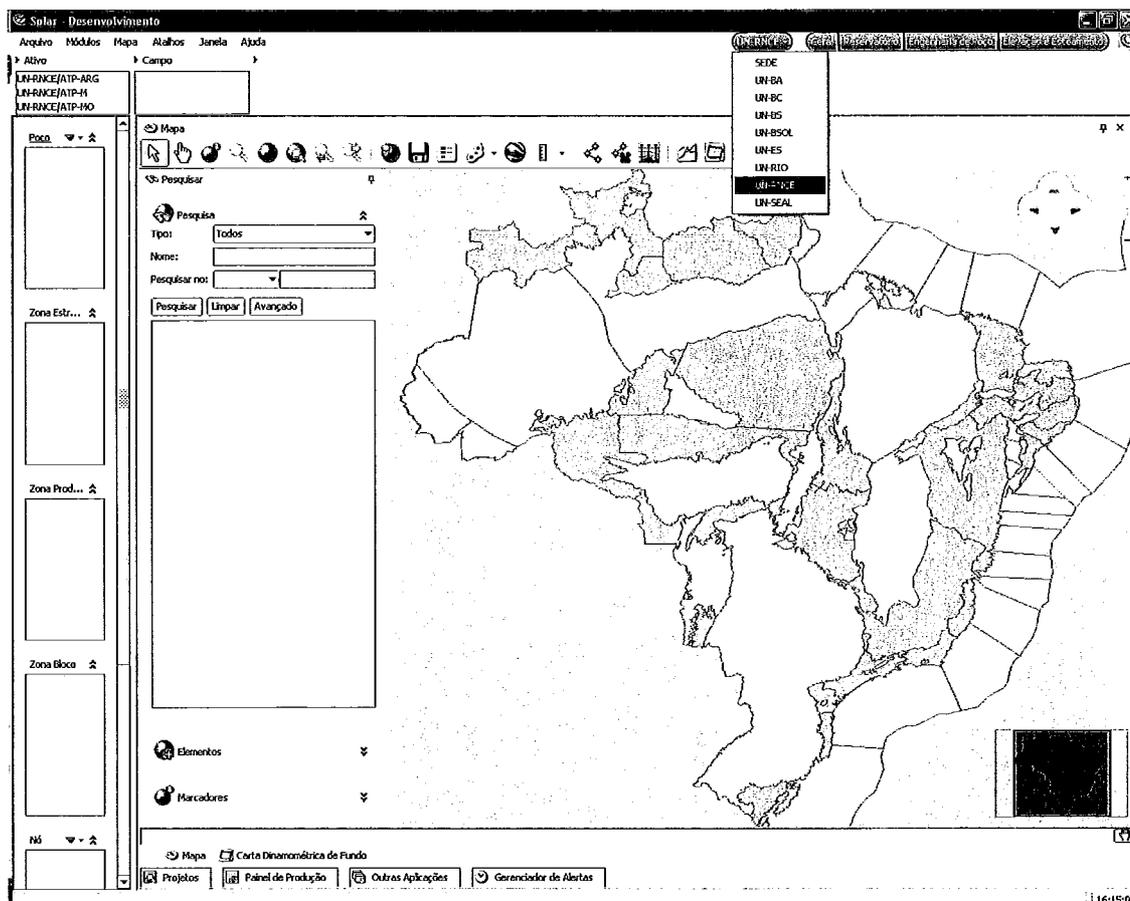


Figura C.1. Visão geral da Interface do SOLAR.

A interface do SOLAR, como pode ser visto na figura C.2, é dividida em quatro partes principais:

- Menu Principal – Para seleção dos módulos e das ferramentas disponíveis no SOLAR;
- Navegador – Para navegação hierárquica pelos elementos de exploração e produção, tais como *campo*, *poço*, *zona de produção* etc;
- Seleção da UN e Perspectiva do Usuário – Para seleção da UN (Unidade de Negócio) e da perspectiva do usuário;

- Seção dos Módulos – Seção destinada aos módulos selecionados pelo usuário.

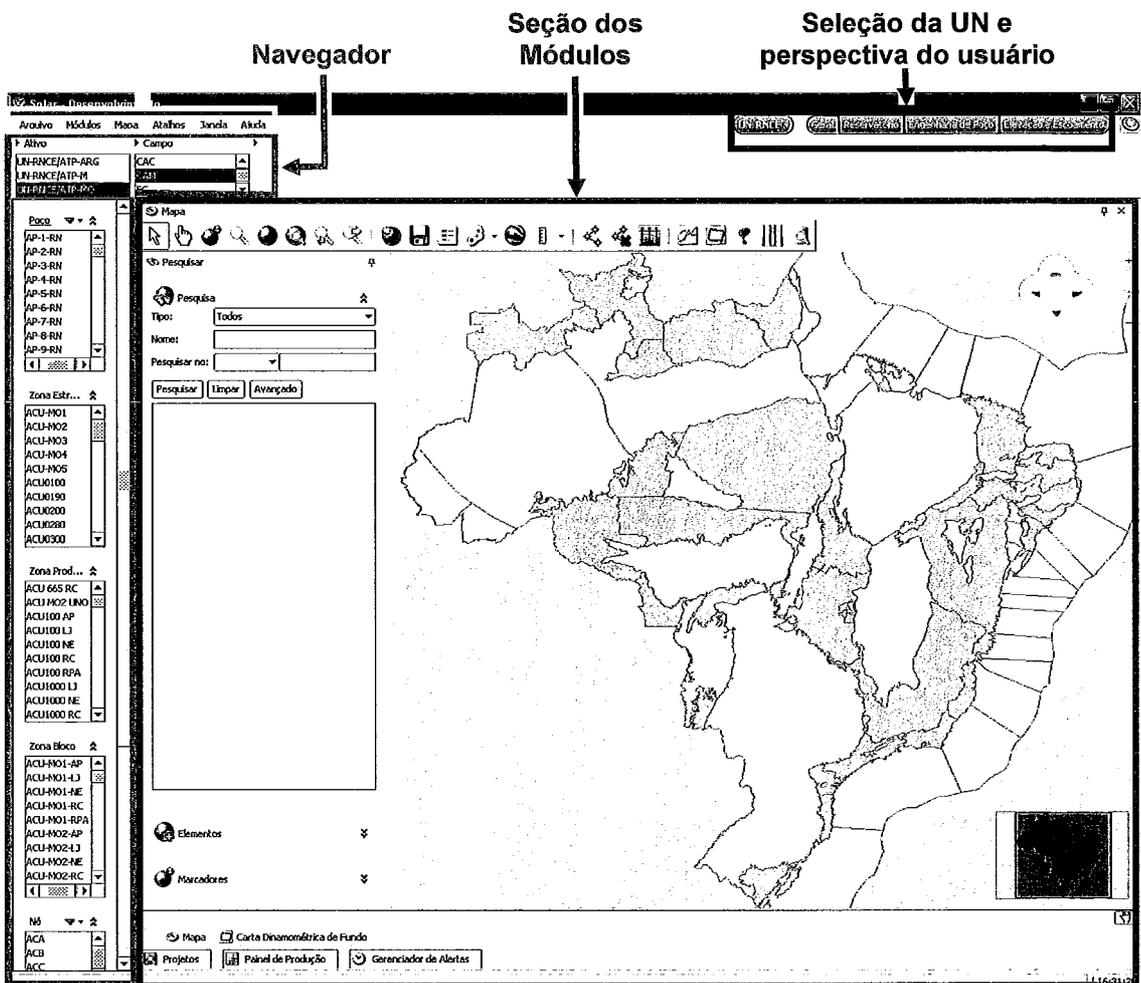


Figura C.2. Interface do SOLAR dividida por áreas.

O IFBM, assim como a maioria dos módulos do SOLAR, é dividido em três partes principais, como pode ser visto na figura C.3:

- Barra de Ferramentas – Para seleção rápida das principais funcionalidades do módulo;
- Seção de Parâmetros – Seção onde ficam os parâmetros selecionados para uso do módulo;
- Seção de Resultados – Seção para visualização e manipulação do resultado fornecido pelo módulo para os parâmetros informados.

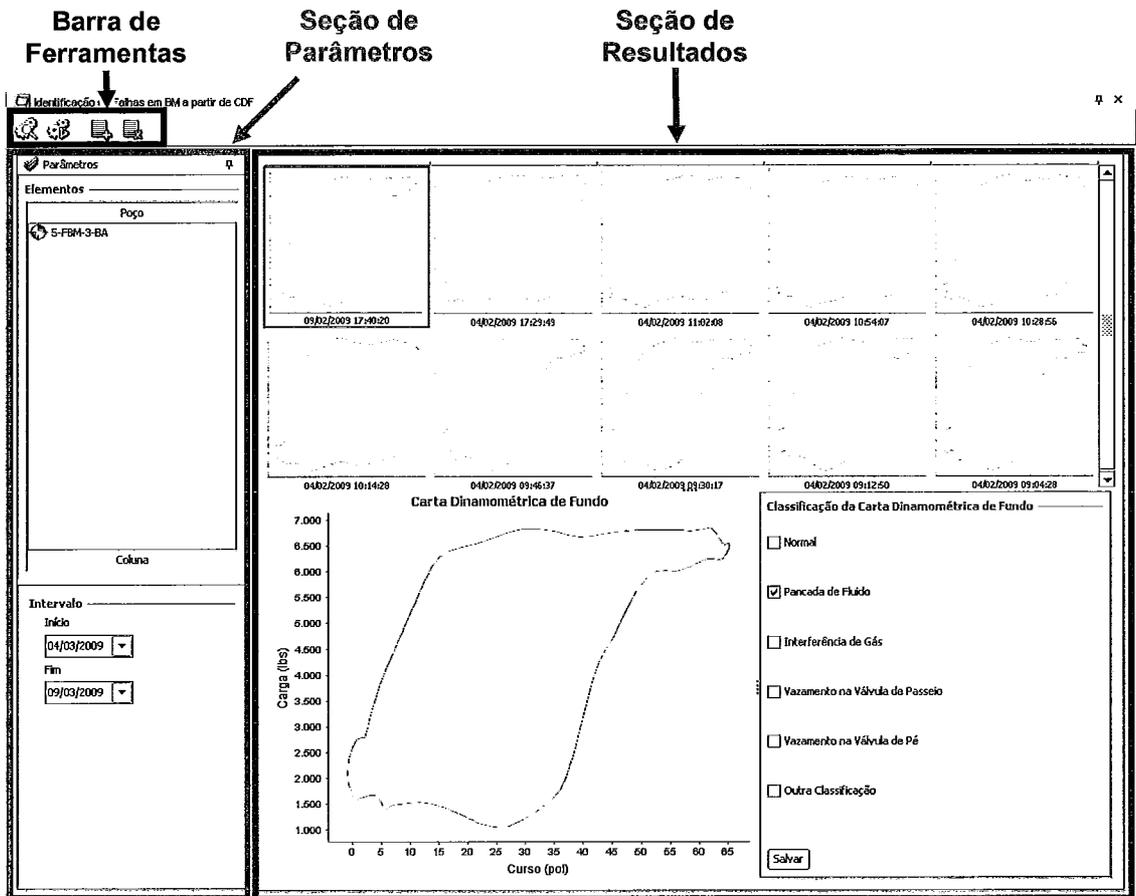


Figura C.3. Interface do módulo dividido por áreas.

No caso do módulo IFBM, a seção de parâmetros espera um poço (ou poço-coluna) e um intervalo de tempo dentro do qual são selecionadas as cartas daquele poço. As principais funcionalidades presentes na barra de ferramenta são:

-  - Visualização das CDFs do poço selecionado;
-  - Classificação, realizada pela rede neural, das CDFs selecionadas;
-  - Inclusão de elementos (poço ou poço-coluna) na seção de parâmetros. A inclusão também pode ser feita pelo navegador SOLAR;
-  - Eliminação de elementos (poço ou poço-coluna) da seção de parâmetros.