UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA DE TELEINFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE TELEINFORMÁTICA

CLÁUDIO MARQUES DE SÁ MEDEIROS

UMA CONTRIBUIÇÃO AO PROBLEMA DE SELEÇÃO DE MODELOS NEURAIS USANDO O PRINCÍPIO DE MÁXIMA CORRELAÇÃO DOS ERROS

FORTALEZA
2008

CLÁUDIO MARQUES DE SÁ MEDEIROS

UMA CONTRIBUIÇÃO AO PROBLEMA DE SELEÇÃO DE MODELOS NEURAIS USANDO O PRINCÍPIO DE MÁXIMA CORRELAÇÃO DOS ERROS

Tese submetida à Coordenação do Curso de Pós-Graduação em Engenharia de Teleinformática, da Universidade Federal do Ceará, como requisito parcial para obtenção do grau de Doutor em Engenharia de Teleinformática.

Àrea de concentração: Sinais e Sistemas

Orientador: Prof. Dr. Guilherme de Alencar Barreto

FORTALEZA

2008

Agradecimentos

Aos meus pais, José de Sá Medeiros e Francisca Neodemia Marques Medeiros, por terem criado as condições para que eu chegasse onde cheguei.

Ao Centro Federal de Educação Tecnológica do Ceará (CEFET-Ce), e especialmente aos professores do Curso de Mecatrônica, por acreditarem e investirem no meu aperfeiçoamento profissional.

Aos professores e funcionários do Departamento de Engenharia de Teleinformática que de forma direta ou indireta participaram do desenvolvimento deste trabalho.

Aos colegas de laboratório (LATIN), por estarem sempre prontos a ajudar, proporcionando excelente ambiente de trabalho.

Aos colegas de trabalho do laboratório de microcontroladores (LABOMICRO) do CEFET, Profs. Luiz Francisco Coelho Coutinho e Carlos Gustavo Castelo Branco, por trabalharem no desenvolvimento da bancada de testes, e ao Prof. Clayton Ricarte, pela colaboração nas simulações computacionais do motor de indução.

Ao Sr. Jacob Tavares Neto, representante local da WEG, e à própria WEG pela doação de motores para a confecção da bancada de testes.

Ao meu orientador, Prof. Dr. Guilherme de Alencar Barreto, por exercer com dignidade sua função e pela confiança depositada.

Em especial à Danyelle, minha mulher, pela paciência, compreensão, incentivo e apoio incondicional.

Resumo

Propõe-se nesta tese um método de poda de pesos para redes Perceptron Multicamadas (MLP). Técnicas clássicas de poda convencionais, tais como *Optimal Brain Surgeon*(OBS) e *Optimal Brain Damage*(OBD), baseiam-se na análise de sensibilidade de cada peso da rede, o que requer a determinação da inversa da matriz Hessiana da função-custo. A inversão da matriz Hessiana, além de possuir um alto custo computacional, é bastante susceptível a problemas numéricos decorrentes do mal-condicionamento da mesma. Métodos de poda baseados na regularização da função-custo, por outro lado, exigem a determinação por tentativa-e-erro de um parâmetro de regularização.

Tendo em mente as limitações dos métodos de poda supracitados, o método proposto baseia-se no "Princípio da Máxima Correlação dos Erros" (MAXCORE). A idéia consiste em analisar a importância de cada conexão da rede a partir da correlação cruzada entre os erros em uma camada e os erros retropropagados para a camada anterior, partindo da camada de saída em direção à camada de entrada. As conexões que produzem as maiores correlações tendem a se manter na rede podada. Uma vantagem imediata deste procedimento está em não requerer a inversão de matrizes, nem um parâmetro de regularização.

O desempenho do método proposto é avaliado em problemas de classificação de padrões e os resultados são comparados aos obtidos pelos métodos OBS/OBD e por um método de poda baseado em regularização. Para este fim, são usados, além de dados artificialmente criados para salientar características importantes do método, os conjuntos de dados bem conhecidos da comunidade de aprendizado de máquinas: Iris, Wine e Dermatology. Utilizou-se também um conjunto de dados reais referentes ao diagnóstico de patologias da coluna vertebral. Os resultados obtidos mostram que o método proposto apresenta desempenho equivalente ou superior aos métodos de poda convencionais, com as vantagens adicionais do baixo custo computacional e simplicidade. O método proposto também mostrou-se bastante agressivo na poda de unidades de entrada (atributos), o que sugere a sua aplicação em seleção de características.

Palavras-chave: Redes Neurais Artificiais, Métodos de Poda, Seleção de Modelos, Correlação de Erros, Seleção de Características.

Abstract

This thesis proposes a new pruning method which eliminates redundant weights in a multilayer perceptron (MLP). Conventional pruning techniques, like Optimal Brain Surgeon (OBS) and Optimal Brain Damage (OBD), are based on weight sensitivity analysis, which requires the inversion of the error Hessian matrix of the loss function (i.e. mean squared error). This inversion is specially susceptible to numerical problems due to poor conditioning of the Hessian matrix and demands great computational efforts. Another kind of pruning method is based on the regularization of the loss function, but it requires the determination of the regularization parameter by trial and error.

The proposed method is based on "Maximum Correlation Errors Principle" (MAXCORE). The idea in this principle is to evaluate the importance of each network connection by calculating the cross correlation among errors in a layer and the back-propagated errors in the preceding layer, starting from the output layer and working through the network until the input layer is reached. The connections which have larger correlations remain and the others are pruned from the network. The evident advantage of this procedure is its simplicity, since matrix inversion or parameter adjustment are not necessary.

The performance of the proposed method is evaluated in pattern classification tasks and the results are compared to those achieved by the OBS/OBD techniques and also by regularization-based method. For this purpose, artificial data sets are used to highlight some important characteristics of the proposed methodology. Furthermore, well known benchmarking data sets, such as IRIS, WINE and DERMATOLOGY, are also used for the sake of evaluation. A real-world biomedical data set related to pathologies of the vertebral column is also used. The results obtained show that the proposed method achieves equivalent or superior performance compared to conventional pruning methods, with the additional advantages of low computational cost and simplicity. The proposed method also presents efficient behavior in pruning the input units, which suggests its use as a feature selection method.

Keywords: Artificial Neural Networks, Pruning Methods, Model Selection, Error Correlation, Feature Selection.

Lista de Figuras

Lista de Tabelas

Lista de Símbolos

$C_{hi}[i,j]$	índice de correlação de erros entre o neurônio i da camada oculta e unidade j da camada de entrada
$C_{oh}[k,i]$	índice de correlação de erros entre o neurônio k da camada de saída e o neurônio i da camada oculta
CR_{train}	taxa de classificação no conjunto de treinamento
$d_k(t)$	saída desejada (rótulo) do neurônio k da camada de saída
$d(x,y,\theta)$	função custo com termo de regularização
$e_k^{(o)}$	erro gerado pelo neurônio k da camada de saída
$e_i^{(h)}$	sinal de erro retropropagado para o neurônio i da camada escondida
$\mathbf{E}_h$	matriz das projeções dos erros na camada oculta
$\mathbf{E}_i$	matriz das projeções dos erros na camada de entrada
$\mathbf{E}_o$	matriz dos erros de saída
$f(x,\theta)$	função de x dado o conjunto de parâmetros θ
$g(\theta)$	conjunto de funções admissíveis
I^n	cubo unitário n-dimensional
J_{train}	índice usado para avaliação do desempenho da rede no conjunto de dados de treinamento
J_{tol}	índice de avaliação definido pelo usuário
$\hat{L}$	verossimilhança estimada
M	número de neurônios na camada de saída
m_{ki}	peso associado à ligação entre neurônio i da camada intermediária e saída k de uma rede supervisionada
N_h	número de neurônios na camada escondida
n_θ	número de pesos do modelo
P	dimensão do vetor de entrada de uma rede neural
$p(y\|x,\theta)$	função distribuição condicional, onde θ é o vetor de parâmetros que especifica a rede
Q	número de neurônios na camada escondida
$q(x)$	função de distribuição de probabilidade desconhecida
$q(x,y)$	função de distribuição conjunta

$q(y\|x)$	função distribuição condicional
$r(\theta)$	termo de regularização
$s(x, y, \theta)$	funcional ou função de custo
t	índice indicativo de amostra apresentada ao algoritmo de retropropagação
$u_i^{(h)}$	ativação do neurônio i da camada escondida de uma rede supervisionada
$u_k^{(o)}$	ativação do neurônio k da camada de saída de uma rede supervisionada
W	número de pesos de uma rede neural
w_{ij}	peso associado à ligação entre entrada j e neurônio i da camada intermediária de uma rede supervisionada
x_j	atributo j do vetor de entrada
y'	predição feita por um modelo
$y_i^{(h)}$	saída do neurônio i da camada escondida de uma rede supervisionada
$y_k^{(o)}$	saída do neurônio k da camada de saída de uma rede supervisionada
Θ	subconjunto do espaço vetorial $\Re^n$
θ	vetor de parâmetros que especifica a rede
$\theta_i^{(h)}$	limiar (*bias*) do neurônio i da camada escondida
$\theta_k^{(o)}$	limiar de ativação do neurônio da saída k
$\varphi_i(\cdot)$	função de ativação sigmoidal
ϵ	erro relativo
$\delta_k^{(o)}$	gradiente local do neurônio k de saída
$\delta_i^{(h)}$	gradiente local do neurônio i da camada escondida
η	taxa de aprendizagem das redes neurais artificiais
ε_{train}	erro quadrático médio por época de treinamento

Lista de Siglas

RNAs	*Redes Neurais Artificiais*
MSE	*Mean-Squared Error*
MLP	*Multilayer Perceptron*
GCV	*Generalized Cross-Validation*
FPE	*Final Prediction Error*
PSE	*Predicted Squared Error*
FIS	*Final Information Statistics*
IQ	*Information Quantity*
MLL	*Mean Log-Likelihood*
MDL	*Minimum Description Length*
NIC	*Network Information Criterion*

Sumário

1 Introdução

1.1 Introdução

Desde a introdução da idéia de usar redes neurais como dispositivos computacionais nos anos 40, pesquisadores têm desenvolvido e/ou adaptado teorias para explicar o comportamento complexo de tais dispositivos. Especialmente nos anos 80, quando o algoritmo de retropropagação teve grande atenção da comunidade científica, muitos esforços têm sido empregados na consolidação de uma teoria de redes neurais *feedforward*, o que aumenta a credibilidade das redes neurais como ferramenta tecnológica.

Neste sentido, as redes neurais artificiais (RNAs) têm sido objeto de muita atenção tanto no meio acadêmico, onde os pesquisadores têm explorado a fundo o potencial desta ferramenta, quanto por profissionais de orientação prática, os quais têm aplicado as RNAs em soluções de problemas tecnológicos, notadamente em ambientes industriais.

Dentre os vários tipos de redes neurais, a rede perceptron multicamadas (*MultiLayer Perceptron* - MLP) tem recebido muita atenção. Isso é motivado principalmente pelo fato de que pesquisadores provaram que a MLP com uma camada escondida é um aproximador universal de função (HORNIK et al., 1989; CYBENKO, 1989; FUNAHASHI, 1989), isto é, ela pode aproximar qualquer função contínua com uma dada precisão, desde que contenha neurônios suficientes. A questão então é: quantos neurônios são considerados como suficientes?

A simples escolha de uma MLP com um número elevado de neurônios parece bastante atrativa, pois é sabido que MLPs grandes geralmente aprendem razoavelmente rápido e apresentam menor sensibilidade a condições iniciais (KROSE; SMAGT, 1993). Além disso, redes grandes têm uma tendência a usar os seus diversos graus de liberdade para mapear com precisão arbitrária a relação entrada-saída subjacente aos dados do conjunto de treinamento. Entretanto, quando submetidas a conjuntos de teste com dados desconhecidos, tais redes geralmente não apresentam bons resultados. Isto é, a rede memoriza os

dados de treinamento ao invés de aprender a verdadeira função de mapeamento entrada-saída e ter a capacidade de generalizar o conhecimento adquirido.

Arquiteturas sobreparametrizadas ainda apresentam alguns inconvenientes. Quando se pretende implementá-las em sistemas embarcados, apresentam dificuldades de implementação, tais como grande demanda de memória e, no caso de aplicações em tempo real, o tempo de processamento pode ser inaceitável. Entretanto, a aplicação de um método de poda de pesos favorece a redução da estrutura, o que pode diminuir demanda por memória e tempo de processamento, além de geralmente melhorar a capacidade de generalização da rede (SIETSMA; DOW, 1988).

Existem vários métodos bem conhecidos e consolidados para a realização de poda em MLPs. Alguns dos quais se utilizam de técnicas lineares, como a análise de componentes principais (PCA), para detectar baixa relevância de atributos ou neurônios ocultos na implementação de mapeamentos entrada-saída de redes neurais (MOODY, 1992; LEVIN; MOODY, 1994; HENRIQUE et al., 2000; CHEN et al., 2001). Outros métodos podam pesos sinápticos individualmente. Mas, com a evolução do processo de poda, pode-se ter a completa eliminação de neurônios. Dentre estes últimos, destacam-se os métodos baseados na estimação da sensibilidade da função de erro em relação à remoção dos pesos (LECUN et al., 1990; HASSIBI et al., 1993), e os métodos que se utilizam da adição de termo de regularização à função de custo, objeto de minimização durante o processo de treinamento (CHAUVIN, 1989; HINTON, 1989; ISHIKAWA, 1990), penalizando a magnitude dos pesos.

Alguns pesquisadores são cautelosos quanto ao uso de técnicas como PCA, por exemplo, que por ser linear não seria adequada para a poda em modelos neurais (não-lineares). Já os métodos baseados na adição de termos de penalização à função de custo, embora possam apresentar bons resultados, são bastante dependentes do ajuste de parâmetros experimentalmente e das condições iniciais de treinamento. Os métodos baseados na estimação de sensibilidades também apresentam bons resultados, mas, em geral, são aplicados sob a suposição de que o treinamento prévio da rede tenha atingido condições específicas.

Neste trabalho é apresentada uma metodologia eficiente de poda de neurônios excedentes em uma MLP previamente treinado sem a necessidade de cálculos complexos ou ajuste de quaisquer parâmetros de regularização. O método proposto é baseado na análise da correlação entre os erros produzidos pelos neurônios de uma dada camada e os erros retropropagados para os neurônios da camada antecessora. Não há qualquer suposição prévia, seja relativa ao processo de treinamento ou à forma da superfície de erro. A apli-

cação sucessiva dessa metodologia de poda de pesos pode levar, eventualmente, à completa eliminação de todas as conexões de alguns neurônios.

1.2 Motivação

Redes MLP treinadas a partir de conjuntos de dados que contenham informação representativa e relevante, em quantidade e qualidade, frequentemente atendem as expectativas de projetistas no que se refere ao desempenho na classificação ou aproximação sobre conjunto de dados de treinamento, desde que o critério de desempenho requerido esteja compatível com a complexidade do problema. Entretanto, os projetistas também almejam bons resultados na generalização sobre conjuntos de teste e validação, o que frequentemente demanda trabalho experimental exaustivo. Notadamente, uma grande dificuldade é estabelecer a topologia da rede mais adequada para a solução do problema em questão.

A motivação desta pesquisa é, principalmente, o desenvolvimento de uma metodologia de poda de rede que, a partir de uma topologia de MLP previamente treinada com foco apenas no bom desempenho da rede sobre o conjunto de dados de treinamento, possa ser aplicada de maneira simples e que confira também, ao final de sua aplicação, bons resultados na generalização sobre o conjunto de dados de teste e validação.

1.3 Objetivos da Pesquisa

O desenvolvimento pretendido de uma metodologia de poda de redes MLP, bem como o reconhecimento destas redes como ferramentas poderosas na modelagem de funções não-lineares, exigem um aprofundamento do conhecimento de sua funcionalidade, bem como das ferramentas matemáticas capazes de melhorar seu desempenho. Sendo assim, são relacionados a seguir os principais objetivos desta pesquisa:

- investigar os limites de classificação da rede MLP, bem como os fatores que influenciam na sua capacidade de generalização;
- investigar as diversas ferramentas utilizadas na seleção de modelos neurais;
- propor uma metodologia com baixo custo computacional que busque a obtenção de MLPs com baixo número de parâmetros e apresentem resultados satisfatórios na generalização sobre dados desconhecidos;

- comparar o desempenho do método proposto com outros já consolidados, enfatizando suas vantagens e limitações;

1.4 Contribuições da Tese

As principais contribuições desta tese estão relacionadas a seguir:

- O desenvolvimento de um método simples para poda de conexões sinápticas redundantes em MLPs sem que haja qualquer suposição prévia sobre a forma da função custo e com extensa experimentação comprobatória de sua eficiência;
- Extensão da aplicação do método de poda proposto para a busca de topologia mínima e para seleção de características;
- A utilização de correlação e análise de componentes principais como ferramentas para seleção de modelos neurais, chamando a atenção para o potencial que a rede MLP dual, linear nos parâmetros, apresenta para aplicação de técnicas lineares.

1.5 Produção Científica

Ao longo do desenvolvimento desta tese os seguintes artigos científicos foram produzidos:

- **Cláudio M. S. Medeiros** & Guilherme A. Barreto (2007), "An Efficient Method for Pruning the Multilayer Perceptron Based on the Correlation of Errors", apresentado na *International Conference on Artificial Neural Networks* (ICANN 2007), 09/09/2007 - 13/09/2007, Porto - Portugal.
- **Cláudio M. S. Medeiros** & Guilherme A. Barreto (2007), "Um Método Eficiente de Poda de Perceptron de Múltiplas Camadas Baseado na Correlação Entre os Erros", apresentado no VIII Simpósio Brasileiro de Automação Inteligente (SBAI 2007), 08/10/2007 - 11/10/2007, Florianópolis-SC.
- **Cláudio M. S. Medeiros** & Guilherme A. Barreto (2007), "Pruning the Multilayer Perceptron Through the Correlation of Backpropagated Errors", apresentado na *Seventh International Conference on Intelligent Systems Design and Applications* (ISDA 2007), 22/10/2007 - 24/10/2007, Rio de Janeiro-RJ.

1.6 Organização Geral da Tese

O restante desta tese está organizado em seis capítulos. Um breve comentário sobre cada um deles é feito a seguir.

O capítulo 2 é iniciado com a apresentação do algoritmo do gradiente descendente para retropropagação de erros, comumente usado no treinamento de MLPs. Em seguida, é apresentado um estudo sobre os limites de classificação do perceptron e a interpretação funcional geométrica de seus componentes. São tratados ainda, aspectos relacionados com a capacidade de generalização da rede MLP.

No capítulo 3 são apresentadas diversas ferramentas utilizadas na seleção de modelos neurais. As ferramentas vão desde a utilização de regras heurísticas até a aplicação de métodos de poda de rede, passando pelo cálculo de índices de avaliação baseados em inferência estatística.

O capítulo 4 é dedicado à apresentação da metodologia proposta para poda de rede, e são apresentados resultados de simulações computaconais comparativos entre o método proposto e outros métodos de seleção de modelos neurais já bem estabelecidos.

No capítulo 5 o método de poda proposto no capítulo anterior é apresentado como uma ferramenta para seleção de características. A utilização de PCA associado à metodologia proposta também é demonstrada com resultados consistentes.

No capítulo 6 são apresentadas as principais conclusões do trabalho e apontadas as suas principais perspectivas para trabalhos futuros.

O apêndice A mostra a forma de implementação de dois dos algoritmos utilizados para efeito de comparação com o CAPE, o OBS e o OBD. É dada ênfase ao cálculo recursivo da inversa da matriz Hessiana utilizada no OBS.

No apêndice B é apresentado um estudo sobre os principais tipos de falhas ocorridas em motores de indução trifásicos e sobre diversas técnicas utilizadas para a detecção destas falhas. Isso, com o intuito de fornecer subsídios para o melhor entendimento da técnica de detecção de curto-circuito entre espiras estatóricas de motor de indução trifásico acionado por conversor de freqüência apresentado no apêndice C. Aqui, o CAPE foi aplicado como ferramenta para seleção de características.

O apêndice D contém os resultados de um estudo comparativo entre os custos computacionais do CAPE e do OBS.

Finalmente, o apêndice E é dedicado às tabelas de características dos conjuntos de dados utilizados nesta tese.

2 *Rede MLP: Fundamentos, Limitações e Heurísticas de Treinamento*

A rede MLP tem se mostrado uma ferramenta poderosa na modelagem de mapeamentos entrada-saída tipicamente encontrados em problemas de aproximação de funções (regressão) e classificação de padrões. A rede MLP é especialmente indicada para modelagem caixa-preta de mapeamentos não-lineares a partir de treinamento supervisionado. Estas características são bastante atrativas, principalmente para pretensos usuários que já elaboraram modelos analíticos/dedutivos complicados.

O aprendizado a partir de exemplos (também chamado de aprendizado indutivo), embora seja uma idéia intuitiva, não pode ser tratada como uma questão banal. Na realidade existem algoritmos bastante sofisticados sendo utilizados para realizar a adaptação dos parâmetros do modelo neural. A implementação de alguns destes algoritmos requer conhecimentos sólidos na área de otimização não-linear. Entretanto, o algoritmo do gradiente descendente para retropropagação do erro se apresenta como uma das alternativas de maior simplicidade e, talvez por isso, seja bastante popular. Porém, esta aparente simplicidade pode ocultar aspectos importantes e decisivos para o sucesso da aplicação da rede MLP.

Isto posto, após a apresentação de uma breve descrição da rede MLP e do algoritmo de retropropagação do erro de saída, são realizadas algumas considerações sobre a interpretação funcional das camadas e dos neurônios em cada camada da MLP e apresentadas algumas recomendações para o treinamento destas redes.

2.1 O Algoritmo de Retropropagação do Erro

Embora se faça alguma menção sobre perceptrons com mais de uma camada escondida neste capítulo, o enfoque desta tese é o perceptron com apenas uma camada escondida. A

Figura 2.1 mostra a topologia em foco nesta breve descrição do algoritmo de retropropagação aplicado ao treinamento de uma MLP completamente conectada com apenas uma camada escondida (HAYKIN, 2002; PRINCIPE et al., 2000).

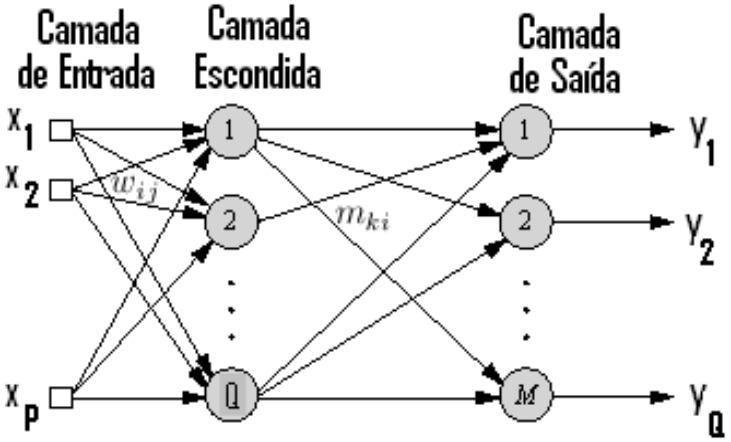

Figura 2.1: MLP com uma camada escondida

No passo t, a ativação de um neurônio da camada escondida é dada por

$$u_i^{(h)}(t) = \sum_{j=1}^{P} w_{ij}(t)x_j(t) - \theta_i^{(h)}(t) = \sum_{j=0}^{P} w_{ij}(t)x_j(t), \quad i = 1, \ldots, Q \tag{2.1}$$

na qual $w_{ij}(t)$ é uma conexão sináptica (peso) entre a j-ésima entrada e o i-ésimo neurônio da camada escondida, $\theta_i^{(h)}(t)$ é o limiar do i-ésimo neurônio da camada escondida, Q $(2 \leq Q < \infty)$ é o número de neurônios da camada escondida e P é a dimensão do vetor de entrada (excluindo o limiar). Definindo $x_0(t) = -1$ e $w_{i0}(t) = \theta_i^{(h)}(t)$ simplifica bastante a notação.

A saída do i-ésimo neurônio oculto é então dada por

$$y_i^{(h)}(t) = \varphi_i\left[u_i^{(h)}(t)\right] = \varphi_i\left[\sum_{j=0}^{P} w_{ij}(t)x_j(t)\right], \tag{2.2}$$

cujo $\varphi_i(\cdot)$ é geralmente uma função sigmoidal. Similarmente, os valores de saída dos neurônios da camada de saída são dados por

$$y_k^{(o)}(t) = \varphi_k\left[u_k^{(o)}(t)\right] = \varphi_k\left[\sum_{i=0}^{Q} m_{ki}(t)y_i^{(h)}(t)\right], \tag{2.3}$$

em que $m_{ki}(t)$ é o peso da conexão sináptica entre o i-ésimo neurônio da camada escondida e o k-ésimo neurônio $(k = 1, \ldots, M)$ da camada de saída, e $M \geq 1$ é o número de neurônios de saída. Novamente, com o propósito de simplificar a notação, define-se $y_0(t) = -1$ e

$m_{k0}(t) = \theta_k^{(o)}(t)$, em que $\theta_k^{(o)}(t)$ é o limiar do neurônio da saída k.

É importante salientar que durante o processo de treinamento os pesos estão sendo ajustados a cada apresentação de vetor de entrada, e por isso, nesta notação estão em função de t.

O caminho reverso se dá pela retropropagação dos sinais de erro de saída através das camadas de saída e escondida(s), até atingir a camada de entrada. Para isso é necessário inicialmente calcular o valor do erro $e_k^{(o)}(t)$ gerado por cada neurônio de saída no passo corrente t

$$e_k^{(o)}(t) = d_k(t) - y_k^{(o)}(t), \quad k = 1, \ldots, M \tag{2.4}$$

em que $d_k(t)$ é o valor desejado para a saída do k-ésimo neurônio da camada de saída. A retropropagação do sinal de erro $e_k^{(o)}(t)$ do k-ésimo neurônio de saída através de sua função de ativação é obtida pelo produto deste mesmo sinal de erro com a derivada $\varphi_k'\left[u_k^{(o)}(t)\right] = \partial\varphi_k/\partial u_k^{(o)}$, obtendo-se assim o *gradiente local* do k-ésimo neurônio de saída

$$\delta_k^{(o)}(t) = \varphi_k'\left[u_k^{(o)}(t)\right] e_k^{(o)}(t). \tag{2.5}$$

Similarmente, o gradiente local $\delta_i^{(h)}(t)$ do i-ésimo neurônio da camada escondida é dado por

$$\delta_i^{(h)}(t) = \varphi_i'\left[u_i^{(h)}(t)\right] \sum_{k=1}^{M} m_{ki}(t)\delta_k^{(o)}(t) = \varphi_i'\left[u_i^{(h)}(t)\right] e_i^{(h)}(t), \quad i = 0, \ldots, Q, \tag{2.6}$$

em que o termo $e_i^{(h)}(t)$ pode ser considerado como o sinal de erro *retropropagado* ou *projetado* para o i-ésimo neurônio da camada escondida, desde que tais "sinais de erro da camada escondida" são combinações lineares dos "verdadeiros" sinais de erro cometidos nos neurônios da camada de saída.

Finalmente, os pesos sinápticos dos neurônios de saída são atualizados de acordo com a seguinte regra

$$m_{ki}(t+1) = m_{ki}(t) + \eta\delta_k^{(o)}(t)y_i^{(h)}(t), \quad i = 0, \ldots, Q, \tag{2.7}$$

na qual $0 < \eta < 1$ é a taxa de aprendizagem. Os pesos dos neurônios da camada escondida, por sua vez, são também ajustados similarmente pela regra de aprendizagem

$$w_{ij}(t+1) = w_{ij}(t) + \eta\delta_i^{(h)}(t)x_j(t), \quad j = 0, \ldots, P. \tag{2.8}$$

Uma apresentação completa de todos os N padrões do conjunto de treinamento du-

rante o processo de treinamento é chamada de *época*. Muitas épocas podem ser necessárias até que haja convergência na aplicação do algoritmo de retropropagação. Assim, uma boa prática é apresentar aleatoriamente os dados do conjunto de treinamento, época por época, com o objetivo de tornar estocástica a busca no espaço de pesos durante o processo de treinamento.

Uma forma simples de avaliar a convergência é através do erro quadrático médio

$$\varepsilon_{train} = \frac{1}{2N}\sum_{t=1}^{N}\sum_{k=1}^{M}\left[e_k^{(o)}(t)\right]^2 = \frac{1}{2N}\sum_{t=1}^{N}\sum_{k=1}^{M}\left[d_k(t) - y_k^{(o)}(t)\right]^2, \tag{2.9}$$

calculado ao fim de cada rodada de treinamento usando os vetores de dados de treinamento. Durante este processo, se a variação do erro estiver abaixo de um valor predeterminado, ou mesmo se o erro cai abaixo de um valor predeterminado, considera-se que houve convergência.

A saída do k-ésimo neurônio da camada de saída da rede treinada é dada por

$$y_k^{(o)}(t) = \varphi_k\left[\sum_{i=0}^{Q} m_{ki}\varphi_i\left(\sum_{j=0}^{P} w_{ij}x_j(t)\right)\right]. \tag{2.10}$$

É importante notar que os pesos da Equação (2.10) são fixos.

Antes que o usuário aplique a rede MLP para solucionar um determinado problema, recomenda-se que a rede seja submetida a uma avaliação do seu desempenho sobre um conjunto de dados, contendo amostras jamais apresentadas anteriormente à rede, chamado de conjunto de teste. Este procedimento é comumente conhecido como avaliação da capacidade de generalização da rede.

2.2 Fundamentos da Rede MLP

É sabido que um perceptron simples aplicado a classificação pode resolver apenas problemas com classes linearmente separáveis (classes podem ser separadas por um hiperplano) (MINSKY; PAPERT, 1988) e que, por extensão, se a função de ativação de cada neurônio numa rede de múltiplas camadas for linear, então esta rede também será capaz de resolver apenas problemas linearmente separáveis (BOSE; GARGA, 1993).

Problemas que não são linearmente separáveis podem ser resolvidos com redes multicamadas cujos neurônios possuem funções de ativação não lineares. Então, a questão

é: quantas camadas escondidas devem ser usadas e quantos neurônios em cada camada escondida?

De Figueiredo (1980) mostrou que o teorema de Kolmogorov poderia ser utilizado em reconhecimento estatístico de padrões. Posteriormente Hecht-Nielsen (1987) propôs que qualquer função contínua definida num cubo unitário n-dimensional (I^n) poderia ser implementada por uma rede MLP com apenas uma camada escondida com $2n+1$ unidades. Entretanto, ele demonstra esta propriedade para neurônios equipados com funções de ativação não lineares com alta complexidade (não sigmoidais). Por isso, Poggio & Girosi (1989) não consideraram relevante o resultado de *Kolmogorov* para redes neurais.

Hornik et al. (1989), Cybenko (1989) e Funahashi (1989) com resultados similares, provaram de forma não construtiva que qualquer função contínua pode ser aproximada arbritrariamente bem por uma rede MLP com apenas uma camada escondida com neurônios escondidos semilineares usando limiares (*bias*) e um neurônio linear na saída.

Maiorov & Pinkus (1999), tratando a rede MLP como um aproximador de funções, afirmam que o grau de aproximação de uma MLP com apenas uma camada escondida com Q neurônios é limitado por baixo pelo grau de aproximação da combinação linear de Q funções rígidas[1]. Eles provaram que, usando funções de ativação sigmoidais estritamente monotônicas, pode-se aproximar arbitrariamente bem qualquer função contínua dentro de um determinado intervalo por uma rede MLP com duas camadas escondidas com um número finito de neurônios em cada camada.

Esses teoremas e provas influenciaram muitos pesquisadores a continuar a trabalhar intensivamente e acreditar nas potencialidades da MLP. Entretanto, o efeito sobre os usuários principiantes, aqueles que vêem a MLP como uma caixa preta, foi o de encorajá-los a simplesmente escolher um número considerado grande de neurônios na camada escondida e tentar resolver seus problemas práticos. Por esta razão serão apresentados a seguir estudos consolidados sobre o funcionamento interno de uma MLP.

2.2.1 Limitações da Rede MLP com Uma Camada Escondida

Principe et al. (2000) afirmam que um perceptron com M ($M \geq 2$) saídas pode dividir o espaço de entrada em M regiões distintas. Eles supuseram que cada par de regiões no espaço de entrada da rede compartilha um limite comum e que esta dita superfície de decisão é composta por segmentos de superfícies lineares. Cada um destes segmentos leva

[1] Funções rígidas são funções multivariadas da forma $g(a_1x_1 + ... + a_dx_d) = g(\mathbf{a}.\mathbf{x})$, em que g: $\Re \rightarrow \Re$, $a = (a_1, ..., a_d) \in \Re^d$ é uma direção fixa, e $\mathbf{x} = (x_1, ..., x_d) \in \Re^d$ são variáveis.

em consideração a saída de um par de neurônios. Dessa forma, existem $M(M-1)/2$ superfícies de decisão.

Kung & Hwang (1988) sugerem que o número (Q) de neurônios na camada escondida de uma MLP com três camadas deve ser igual ao número de padrões distintos de treinamento. Masahiko (1989) conclui que N padrões de entrada requerem $N-1$ neurônios na camada escondida. Estas estimativas podem levar a redes proibitivamente grandes.

Looney (1996), considerando que uma única camada escondida com Q neurônios (Q hiperplanos) divide o espaço de características em interseções convexas de $2Q$ semi-espaços, afirma que o número de tais regiões convexas está entre $Q-1$ (limite inferior alcançado com hiperplanos paralelos) e 2^Q (limite superior). Então, o espaço de características com K classes linearmente separáveis pode ser separado usando Q neurônios, em que $Q = K-1$ é o limite superior e $Q = log_2 K$ é o limite inferior.

Daqi & Yan (2005) observam que um neurônio é capaz de separar apenas duas classes linearmente separáveis e dois neurônios não mais que cinco categorias corretamente. Eles generalizam esta relação entre o número de neurônios na camada escondida, Q, e o número de classes , K, como $2^Q + 1 \geq K$. Assim, para solucionar problemas com K classes, uma rede MLP com uma única camada escondida contendo pelo menos

$$Q \geq log_2(K-1) \geq 2 \tag{2.11}$$

neurônios na camada escondida é necessário. Eles também afirmam que o número final de neurônios na camada escondida ainda está relacionado com a forma da distribuição das amostras e não com o número de amostras e dimensão dos vetores de entrada.

A seguir é apresentada uma análise geométrica da ação dos neurônios e das camadas de uma MLP, podendo facilitar a compreensão dos limites apresentados.

2.2.2 Interpretação Geométrica de Neurônios e Camadas

Xiang et al. (2005) propõem algumas indicações gerais baseadas na interpretação geométrica de pesos, limiares, camadas escondidas e neurônios em cada camada escondida em problemas de aproximação de funções. Eles sugerem que informações sobre a geometria do problema podem simplificar bastante a escolha da arquitetura a ser usada. Basicamente, a camada escondida de uma MLP fornece os blocos de construção básicos com formas semelhantes a partes da função a ser aproximada. As larguras, alturas e posições desses blocos construtivos podem ser arbitrariamente ajustados pelos pesos e

limiares. Uma MLP com quatro camadas é interpretada simplesmente como uma combinação linear de múltiplas MLPs de três camadas que compartilham os mesmos neurônios na camada escondida mas com neurônios de saída diferentes.

Mesmo quando o espaço de entrada da rede neural cresce, a interpretação geométrica abordada no parágrafo anterior é válida. Entretanto, as formas geométricas básicas de funções de alta dimensionalidade são difíceis de serem determinadas. Conseqüentemente, extrair informação geométrica de funções de alta dimensionalidade a partir de um conjunto de dados disponível é um desafio interessante.

Bose & Garga (1993) usam resultados de geometria computacional para projetar redes neurais em problemas de classificação de padrões. Eles construiram um diagrama de Voronoi [2] e obtiveram a informação necessária para projetar a rede neural a partir disto.

Qualquer célula de Voronoi pode ser implementada por uma rede neural com uma camada escondida e um neurônio na camada de saída. O número de neurônios na camada escondida é igual ao número de hiperplanos que confinam o agrupamento (*cluster*) e apenas um neurônio é suficiente para fazer a operação lógica "E". No caso de regiões não convexas ou mesmo disjuntas, os neurônios de uma segunda camada escondida fazem a operação "E" para cada pequena região convexa e um neurônio por classe faz a operação "OU" das várias pequenas regiões convexas para formar uma região não convexa ou agrupar em uma mesma classe duas regiões disjuntas. O número de neurônios na camada de saída é igual ao número de classes distintas.

Daqi & Yan (2005), ao adotar outro ponto de vista interpretativo para a MLP, chamam a atenção para o fato de que em uma rede com uma única camada escondida, a função primordial dos neurônios da camada escondida com funções de ativação sigmoidais é transformar classes não linearmente separáveis no espaço de entrada em classes linearmente separáveis num espaço expandido por estes neurônios. Cada vetor de pesos associado a um neurônio da camada escondida representa um vetor de base do espaço no qual o vetor de entrada será projetado.

[2] Um diagrama de Voronoi é uma partição do espaço d-dimensional em regiões convexas, chamadas de células de Voronoi, cada uma das quais define uma "região de influência"de um dado ponto no seu interior. Cada célula pode ser definida como a interseção de um número finito de subespaços fechados e é, então, confinado por um número finito de hiperplanos.

2.2.3 Quantas Camadas Escondidas?

Teoricamente, uma MLP pode ter tantas camadas escondidas quanto o projetista desejar. Entretanto, na prática, a escolha geralmente é de uma ou duas camadas. A seguir, são apresentados alguns resultados de estudos sobre esta questão.

Mehrotra et al. (1991) usam uma segunda camada escondida quando o número de classes é grande. Nesses casos, quando se usa apenas uma camada escondida, o número de neurônios pode ser muito grande.

Os resultados de Villiers & Barnard (1992) mostram que não há diferença estatística significante entre redes de uma e duas camadas escondidas quando treinadas de forma adequada. Aquelas com apenas uma camada escondida, entretanto, apresentam bons resultados no reconhecimento de padrões com maior frequência.

Tamura & Tateishi (1997) mostram que redes com uma e duas camadas escondidas são estritamente equivalentes apenas com um número infinito de neurônios. Entretanto, eles mostram que a rede com uma segunda camada escondida pode também promover aproximações arbitrariamente precisas, mas com um número menor de pesos. Eles alertam que "pode ser perigoso simplesmente assumir que uma única camada escondida promoverá naturalmente aproximações adequadas".

Haykin (2002) sugere que, para efeito de eficiência no treinamento, não se deve criar topologias com muitas camadas, pois os pesos das camadas mais próximas da entrada da rede serão treinados mais lentamente. Isto se dá pela atenuação imposta ao erro devido a derivada da não-linearidade de cada neurônio em seu ponto de operação durante a retropropagação. Existem maneiras de amenizar este efeito, tal como o uso de taxas de aprendizagem maiores para neurônios pertencentes a camadas mais próximos da entrada da rede, porém, em geral, criam mais parâmetros a serem ajustados.

Xiang et al. (2005) apresentam dois exemplos baseados em simulações, nos quais fica claro que redes com duas camadas escondidas são mais eficientes que com apenas uma camada escondida em termos do número mínimo de pesos necessários para atingir desempenhos similares. Entretanto, a diferença entre o número mínimo de pesos normalmente não é muito grande, e a MLP com apenas uma camada escondida pode ser uma boa opção de projeto por apresentar-se menos suscetível a atrações para mínimos locas devido a sua estrutura menos complicada (VILLIERS; BARNARD, 1992).

Curry & Morgan (2006) realizaram estudos experimentais e concluíram que a incorporação de novas camadas escondidas à MLP adiciona graus de liberdade extras, podendo

assim prover melhor generalização.

Os resultados dos estudos apresentados nos parágrafos anteriores mostram visões algumas vezes conflitantes sobre a escolha do número de camadas escondidas. Pode-se adotar uma postura mais pragmática, começando com uma arquitetura mais simples (perceptron simples) e, conforme se julgue necessário, adiciona-se novas camadas. Entretanto, a utilização de MLPs com apenas uma camada escondida é uma opção atrativa. Sua estrutura mais simples a torna menos suscetível a atrações para mínimos locais, além de apresentar maior uniformidade na velocidade de estabilização dos pesos durante o processo de treinamento, conforme já citado. E ainda, considerando desempenhos similares, o número de pesos de uma MLP com uma única camada escondida não é muito maior que o necessário para uma MLP com duas camadas escondidas.

2.3 Heurísticas Para o Treinamento

Como foi discutido anteriormente, qualquer função contínua pode ser aproximada tão precisamente quanto se deseje por uma MLP com apenas uma camada escondida, desde que esta camada contenha neurônios suficientes. Em um caso especial, Yu (1992) mostra que o treinamento de uma rede com apenas uma camada escondida com o algoritmo de retropropagação do erro não leva a mínimos locais se o número de neurônios na camada escondida for $N-1$, em que N é o número de amostras diferentes do conjunto de dados de treinamento. Na prática, entretanto, outras características da superfície de erro, tais como ausência de declives acentuados ou presença de platôs podem impor dificuldades para a otimização, pelo menos quando um dos critérios de interesse é "a melhor solução achada num dado tempo prático limite". (LOONEY, 1996) considera que treinamento inadequado, arquitetura inapropriada, forte influência de ruído e dados não separáveis, são os maiores responsáveis por falhas no uso da rede MLP.

2.3.1 Preparação dos Dados

Zhang (2007) alerta que em muitos artigos divulgados na comunidade científica os pesquisadores não levam em consideração aspectos relevantes referentes aos dados. Em muitos casos, os dados são usados como se os mesmos não contivessem erros e como se eles fossem perfeitamente representativos do processo. Esta suposição não é necessariamente válida.

O pré-processamento dos dados e a seleção de características são também muito im-

portantes. Geralmente há a recomendação de pré-processar os dados, tendo em vista que esta ação pode salientar relações importantes e tornar os dados mais uniformes, o que facilita o treinamento e reduz a possibilidade da ocorrência de problemas computacionais. Quanto à seleção de características, é importante ter em mente que a utilização de vetores de dados excessivamente grandes pode contribuir para que atributos irrelevantes e suas interações ocultem os atributos essenciais e dificultem o processo de aprendizado.

Outro aspecto importante é a composição dos conjuntos de dados. Embora métodos estatísticos, tais como regressão linear, possam ter conjuntos de teste com dados da ordem de 50% de todos os dados disponíveis, a maioria das redes neurais não pode prescindir de tanto, sob o risco de não conseguir aprender o mapeamento. Não existe uma regra precisa para tal, embora a divisão arbitrária dos dados deve ser evitada. O mais adequado é que o projetista avalie preliminarmente o desempenho da rede para diversas divisões de dados e determine a mais adequada. É importante também salientar que, embora esteja longe de um consenso, a utilização de conjuntos de treinamento contemplados com iguais números de padrões por classe pode resultar em melhores predições. A composição desbalanceada de representantes de classes pode favorecer o desenvolvimento de tendências no mapeamento entrada-saída durante o treinamento da rede, de tal forma que a mesma apresente desempenhos de classificação significativamente melhores para as classes mais representadas, em detrimento das menos representadas.

A natureza da codificação das saídas desejadas (rótulos) geralmente não é considerada em detalhes (LAWRENCE et al., 1996). A aceitação prévia de desvios em relação a função de saída pode levar a modelos ditos como "suaves". Nestes casos, os pesos são menos propícios a serem levados a grandes valores e, conseqüentemente, os neurônios são menos suscetíveis à saturação. Desta forma, é esperado que a aproximação da função de saída seja suave e que o desempenho na generalização seja melhor.

2.3.2 Valores Iniciais dos Pesos

Não existem métodos analíticos para achar os pesos ótimos para uma rede neural, desta forma torna-se necessário o emprego de técnicas iterativas de otimização local ou global. O sucesso destas técnicas é estritamente dependente das condições iniciais (ERDOGMUS et al., 2005).

Uma prática comum é treinar a rede neural a partir de diferentes condições iniciais e escolher a rede que apresente melhor desempenho na generalização.

Mesmo se o processo de treinamento for interrompido numa iteração específica em duas diferentes rodadas de treinamento sobre o mesmo conjunto de dados, a iniciação aleatória dos pesos cria trajetórias diferentes de adaptação dos pesos. Assim, conjuntos diferentes de pesos iniciais podem levar a comportamentos drasticamente diferentes no processo de convergência (LI et al., 1993; SCHMIDT et al., 1993).

É comum escolher aleatoriamente os pesos iniciais uniformemente distribuídos numa faixa específica de valores ($-K$ *a* K). É importante levar em consideração que se K cresce, a complexidade da função de mapeamento entrada-saída é aumentada e a solução encontrada vem a ser significantemente pior (LAWRENCE et al., 1996). Superfícies de erro produzidas por MLPs podem ter muitas regiões de pequenas inclinações em múltiplas dimensões (HECHT-NIELSEN, 1990). Isto é um resultado típico quando um ou mais neurônios apresentam ativações com valores altos, o que causa uma certa insensibilidade das saídas destes mesmos neurônios a pequenas mudanças nos seus respectivos pesos. O neurônio está operando nos extremos da função sigmoidal, onde a inclinação é pequena. Com o crescimento de K, a solução ótima requer que os neurônios operem em regiões com baixas inclinações com maior frequência

Principe et al. (2000) sugerem que a definição da variância da distribuição aleatória dos pesos iniciais seja baseada no número de entradas (*fan-in*) de cada neurônio. Para não-linearidades sigmoidais, acredita-se que esta estratégia aleatória de inicialização faça com que os sinais (líquidos) de entrada de cada neurônio se mantenham na região de linearidade aproximada da função sigmoidal. Para um melhor desempenho no treinamento é desejável fazer com que todos os neurônios aprendam aproximadamente à mesma taxa (HAYKIN, 2002).

Se for possível escolher os pesos iniciais de tal forma que a função de erro já esteja próxima de um mínimo global, então o algoritmo de retropropagação pode levar a rede a um ponto ótimo rapidamente. Neste sentido, Erdogmus et al. (2005) propõem uma metodologia para retropropagar a resposta desejada dos neurônios para a camada de entrada. Este algoritmo leva em consideração a derivada da não-linearidade no ponto de operação do valor corrente da saída desejada, e então resolve analiticamente um problema de regressão linear para cada camada de pesos. Embora o procedimento não imponha explicitamente a redução do erro quadrático médio, foi observado em numerosas simulações que este algoritmo leva a rede MLP a atingir erros quadráticos médios pequenos (muito próximos ao ótimo global).

Vários autores propõem soluções lineares para a atribuição de valores iniciais dos

pesos. Porém, vale ressaltar que estratégias baseadas em métodos lineares não necessariamente levam a uma solução global. Em muitos casos a solução linear é exatamente a que deve ser evitada por conduzir a rede a um forte mínimo local (PRINCIPE et al., 2000).

2.3.3 Implementação do Algoritmo de Otimização

Quanto aos algoritmos de treinamento, o algoritmo do gradiente descendente para a retropropagação do erro é o mais utilizado no treinamento de MLPs. Com ele, não há garantias de que o mínimo global seja atingido. Apesar desta observação teórica, diversas observações experimentais indicam que isto não é um sério problema, desde que seja possível usar algum critério prático de parada do processo de treinamento.

Alguns cuidados heurísticos podem melhorar o desempenho do processo de treinamento com o método do gradiente descendente. Por exemplo, a adição de um valor constante de $0,05$ na derivada da não-linearidade pode reduzir o efeito de paralisia do aprendizado de neurônios cujas funções de ativação estejam operando na região de saturação;

Aspectos importantes estão relacionados com a taxa de aprendizagem. A adoção de taxas de aprendizagem diferenciadas para cada camada da rede, por exemplo, pode equalizar a velocidade do aprendizado da rede, pois os neurônios das camadas mais próximas da entrada aprendem mais lentamente. Assim, pode-se definir taxas de aprendizagem maiores nas camadas mais próximas da entrada (LAWRENCE et al., 1996).

A taxa de aprendizagem também pode ter efeito importante sobre a capacidade de generalização da rede. Saad & Solla (1996) afirmam que o erro de generalização cresce com o crescimento do nível de ruído dos dados de treinamento. Entretanto, eles mostram que a utilização de esquemas que decrescem a taxa de aprendizado ao longo do treinamento pode remover assintoticamente o efeito de ruído aditivo na saída da rede e diminuir o erro de generalização.

Para tornar o processo de treinamento mais robusto em relação a escolha da taxa de aprendizagem e reduzir a oscilação na busca do mínimo da função de erro, o usuário pode lançar mão do termo de momento. Neste caso, os termos $\alpha(m(t) - m(t-1))$ e $\alpha(w(t) - w(t-1))$ devem ser acrescidos às Equações de atualização de pesos (2.7) e (2.8), respectivamente. Bhaya & Kaszkurewicz (2004) demonstraram que o método do gradiente descendente com fator de momento é equivalente ao método do gradiente conjugado para funções de erro quadrático. Ele leva em consideração informação prévia do processo de

otimização e melhora a convergência sem aumentar significantemente a computação.

O usuário também pode contornar a dependência da taxa de aprendizagem com a implementação de métodos avançados de busca e esquemas de otimização global (ERDOGMUS et al., 2005). Entretanto, é importante salientar que métodos baseados no gradiente conjugado, como o método Levenberg-Marquardt por exemplo, aumentam sobremaneira a complexidade do algoritmo de treinamento.

2.3.4 A Arquitetura

Caruana (1993) apresenta um tutorial com resultados relativos a generalização em vários problemas em função do tamanho da rede (número de pesos). Ele varia o número de parâmetros de um número muito pequeno a um número muito grande e percebe que as redes grandes raramente apresentam desempenhos piores que redes pequenas. Ele sugere que "a retropropagação ignora o excesso de parâmetros".

Uma justificativa simples para o fato de que redes maiores podem algumas vezes melhorar os erros de treinamento e de generalização é que os graus de liberdade extras podem ajudar na convergência, isto é, considerando inalterado o número de camadas, a adição de parâmetros extras pode diminuir as chances da rede estacionar em mínimos locais ou platôs (KROSE; SMAGT, 1993).

Setioni & Hui (1993) chamam a atenção para o fato de que redes menores são mais fáceis de treinar, e que a possibilidade de se obter sucesso é maior em poucas rodadas de treinamento.

Todavia, Lawrence et al. (1996) afirmam que em geral, resultados teóricos e empíricos com redes de pequeno porte não superam as de grande porte. Uma razão para isto pode ser atribuída ao efeito de interferência[3]. Para uma rede com muitos neurônios ocultos, esse efeito de interferência é mais pronunciado.

Lawrence et al. (1996) concluem que atingir um mínimo na superfície de erro é relativamente raro e que a rede aprende em tempo razoável, desde que não se use redes com um número de parâmetros maior que o número de dados disponíveis para o treinamento. Eles acreditam que os graus de liberdade no modelo devem ser menores que o número total de dados de treinamento. Entretanto, eles também mostram que uma solução próxima da solução ótima geralmente não é atingida.

[3] Quando os neurônios de uma rede em treinamento não estão saturados, a atualização de cada parâmetro geralmente afeta muitos outros parâmetros.

Além disso, a indicação de que o número de parâmetros deve ser menor que o número de amostras no conjunto de treinamento é tipicamente aceita como verdadeira para a maioria dos conjunto de dados. Entretanto, resultados apresentados em Lawrence et al. (1996) indicam que nem sempre acontece. Em certos casos, o uso de grandes redes melhora o desempenho na generalização. Mas, mesmo assim, permanece desejável achar soluções com o menor número possível de parâmetros, o que pode ser alcançado com algum método de poda.

2.3.5 Sumário de Heurísticas para Treinamento de MLPs

A aprendizagem de MLPs, embora seja simples à primeira vista, requer ajuste de parâmetros e a adoção de algumas estratégias de treinamento. Conforme apresentado nas subseções anteriores, a inobservância de aspectos importantes relacionados com o treinamento pode tornar o processo bastante penoso e com resultados modestos. A seguir são arroladas as principais sugestões e dicas para o treinamento de MLPs:

- Normalizar os dados para a faixa de valores das ativações da rede;
- Aplicar métodos de extração de características;
- Realizar testes preliminares para a indicação do número mais adequado de dados para treinamento e testes;
- Usar a tangente hiperbólica em vez da função logística como função de ativação dos neurônios;
- Evitar o uso dos limites assintóticos (-1 e $+1$) da tangente hiperbólica como rótulos das saídas desejadas. Sugestão: $-0,95$ e $0,95$;
- Adicionar um valor constante de $0,05$ na derivada da não-linearidade para reduzir o efeito de paralisia de neurônios;
- Iniciar os pesos da rede de tal forma que as funções de ativação dos neurônios estejam inicialmente na região linear;
- Adotar taxas de aprendizagem maiores nas camadas mais próximas da entrada da rede numa tentativa de equalizar a velocidade de aprendizado dos neurônios;
- Usar decaimento da taxa de aprendizagem durante o processo de treinamento;

- Usar termo de momento (α) para tornar o processo de treinamento menos oscilatório e menos sensível em relação a escolha da taxa de aprendizagem;
- Escolher topologias que tenham menos pesos a serem ajustados do que dados no conjunto de treinamento;
- Usar processos de aprendizagem baseados em métodos de otimização mais sofisticados, como os baseados no gradiente conjugado (Ex.: método Levenberg-Marquardt);
- Aplicar algum método de poda sobre a rede treinada para eliminar redundâncias e melhorar a capacidade de generalização.

2.4 Conclusões

Neste capítulo foi apresentado inicialmente o algoritmo do gradiente descendente para a retropropagação dos erros. Este é o algoritmo fundamentalmente utilizado nesta tese para o treinamento de MLPs com uma única camada escondida.

Foram tratadas também, mesmo que superficialmente, questões teóricas relativas aos limites e capacidade da MLP. A compreensão e a interpretação funcional de neurônios e camadas na rede também foram abordadas. A idéia é que este tipo de conhecimento pode ajudar na escolha inicial de uma topologia de rede em problemas simples.

Finalmente, foram apresentadas algumas recomendações que podem contribuir sobremaneira para a implementação de processos de treinamento eficientes de MLPs.

A escolha do modelo neural adequado é um fator importante para o sucesso do processo de treinamento, que merece uma atenção especial e, por isso, é tratada com mais detalhes no capítulo a seguir.

3 *Seleção de Modelos em Redes Neurais*

Uma questão importante quando se pretende usar uma rede perceptron de múltiplas camadas é: quantos neurônios devem ser usados em cada camada oculta?

A resposta a esta questão não é simples. Para muitos usuários, os quais vêem a rede MLP como uma caixa preta, a solução para problemas de aproximação de funções e reconhecimento de padrões consiste em escolher um número de neurônios grande o suficiente para obter resultados tidos como aceitáveis. Para esses usuários, projetar e treinar uma rede neural e fazê-la funcionar parece mais uma arte do que uma ciência. Entretanto, sabe-se que o conhecimento sobre as funções de cada camada da rede MLP bem como de seus neurônios, aliado com alguma experiência podem levar o usuário a escolher um número adequado de camadas e neurônios, alcançando soluções eficientes (XIANG et al., 2005).

Neste capítulo são abordadas diversas técnicas para a determinação do número de neurônios de uma MLP.

3.1 Como Escolher o Número Adequado de Neurônios na Camada Escondida

Em muitas situações práticas é impossível ou pelo menos inviável obter exemplos representativos de todas as condições de entrada para uma rede neural. Desta forma, a rede deve aprender e não simplesmente memorizar os padrões de treinamento. Se ela memoriza, sua resposta ao conjunto de dados de treinamento será muito boa, mas poderá falhar bastante no tratamento de novos dados. O grande objetivo que se pretende alcançar com sistemas adaptativos que aprendem com exemplos é generalizar, a partir dos dados de treinamento, para novas entradas (KRUSCHKE, 1989).

Alguns estudos (GUTIERREZ et al., 1989; REED, 1993) evidenciam que o número de

parâmetros a serem ajustados numa rede neural tem influência sobre sua capacidade de generalização. Redes com complexidade reduzida (baixo número de pesos) favorecem a melhoria da generalização. Entretanto, é importante ter em mente que uma rede com poucos parâmetros pode não ser capaz de aprender a partir dos dados, não importando qual algoritmo de treinamento seja aplicado. Por outro lado, redes de grande porte (grande número de pesos) apresentam menor sensibilidade a parâmetros de aprendizagem, condições iniciais e mínimos locais. Entretanto, a velocidade com a qual o mapeamento entrada-saída é realizado pode ser severamente diminuída. Redes de tamanho intermediário também podem aprender vagarosamente e ser muito sensíveis a condições iniciais e parâmetros de treinamento.

Na maioria dos casos, o número de neurônios ocultos é determinado a partir do treinamento de várias topologias e estimação do erro de generalização de cada uma. Nesses casos, uma prática comum é estimar o erro de generalização durante o treinamento e interrompê-lo quando o mesmo começar a crescer. Entretanto, esta técnica, conhecida como parada prematura (*early stopping*), pode não ser prática quando se tem disponível poucos dados para treinamento.

Como já foi mencionado anteriormente, a complexidade da função a ser aproximada ou da classificação a ser aprendida, a quantidade de ruído incorporado aos dados, o tipo de função de ativação dos neurônios e o algoritmo de treinamento têm grande influência na capacidade de generalização de uma rede neural. A seguir, serão apresentados alguns métodos ou regras para a escolha ou indicação do número de neurônios.

3.2 Regras Heurísticas

Muitos pesquisadores investigaram e ainda investigam o problema de determinar o número de neurônios mais adequado a ser usado em uma rede neural. Baseados em observações sobre exaustivas experimentações ou em simplificações de métodos bastante elaborados, alguns sugerem regras ou dicas simples como ponto de partida. Algumas destas regras e dicas serão apresentadas nesta seção.

Redes neurais, assim como qualquer classificador treinado não parametricamente, requerem muitos dados para que se tenha um treinamento apropriado, pois não se faz suposições estatísticas prévias sobre os dados. Assim, o tamanho do conjunto de dados de treinamento influi diretamente no desempenho dessa classe de máquinas de aprendizado. De acordo com Baum & Haussler (1989), uma rede neural contendo apenas uma camada

escondida e usada como classificador binário, quase sempre apresentará bom desempenho na generalização desde que duas condições sejam satisfeitas:

- O erro relativo cometido no conjunto de treinamento seja menor que $\epsilon/2$;
- O número de exemplos, N, usados no treinamento seja

$$N \geq \frac{32W}{\epsilon} \ln\left(\frac{32W}{\epsilon}\right) \tag{3.1}$$

onde "ln" denota o logaritmo natural, W o número de pesos na rede e ϵ é um parâmetro de exatidão.

Haykin (2002), ignorando o fator logarítmico na Equação (3.1), sugere que o número apropriado de amostras de treinamento é, com uma aproximação de primeira ordem, diretamente proporcional ao número de pesos na rede e inversamente proporcional ao erro ϵ. Assim, o número de padrões de treinamento (N) requeridos para classificar exemplos de teste com um erro relativo igual a ϵ é aproximadamente dado por

$$N > \frac{W}{\epsilon}. \tag{3.2}$$

Esta equação sugere que o número requerido de padrões de treinamento cresce linearmente com o número de parâmetros livres da rede MLP. Uma boa dica é $N \sim 10W$ ($\epsilon = 0,1$). Entretanto, é suposto que o conjunto de dados de treinamento é representativo de todas as condições encontradas no conjunto de teste.

O projetista deve tomar cuidado com estas regras que simplesmente associam o número de pesos ao número de amostras do conjunto de treinamento. Tais regras revelam uma preocupação apenas com o sobreajustamento (*overfitting*). Entretanto, algumas vezes, se dispõe de um número muito grande (não se sabe o quão grande) de exemplos de treinamento em relação ao número de parâmetros a ser ajustado, mas estes exemplos não representam todas as regiões da função a ser mapeada de forma quantitativamente e/ou qualitativamente equilibrada. Assim o mapeamento entrada-saída implementado pela rede treinada pode ser tão suave em regiões de pouca representatividade dos dados que ocorra subajustamento (*underfitting*) ao invés de sobreajustamento. Ainda mais, na medida do possível, o projetista deve levar em consideração o ruído contido nos dados, pois em casos onde não se tem ruído pode ser necessário um número de dados que seja apenas o dobro do número de pesos para que se evite o sobreajustamento. Por outro lado, em casos bastante ruidosos, dados em número 30 vezes maior que pesos podem não ser suficientes (FAQ, 2006).

Alguns pesquisadores supõem que o número de neurônios ocultos também está relacionado com a dimensão do vetor de entrada (DRAGHICI, 2002; GORMANN; SEJNOWSKI, 1988; GORI; SCARSELLI, 1998). Swingler (1996) e Berry & Linoff (1997) afirmam que jamais será necessário ter neurônios escondidos em número maior que duas vezes o número de características de entrada. Gori & Scarselli (1998) concluem que MLPs com um número de neurônios ocultos menor ou igual à dimensão do espaço de entrada são incapazes de modelar com precisão superfícies de separação.

Por outro lado, foi observado que redes treinadas com algoritmos de retropropagação algumas vezes generalizam melhor quando contêm camadas escondidas com um número de neurônios consideravelmente menor que o da camada precedente. Kruschke (1989) estuda as propriedades funcionais desse tipo de rede e descreve um método para criá-las dinamicamente, durante o aprendizado por retropropagação. Vários autores sugeriram explicações para a melhora na generalização causada por este tipo de configuração da rede. A explicação tem duas vertentes. A primeira, considerando a complexidade, é apresentada por Wieland & Leighton (1987). Eles sugerem que "considerando o aprendizado como aproximação de funções nos leva a ver que a generalização é simplesmente o efeito de uma boa interpolação não-linear dos dados". A argumentação é que melhor generalização vem de mapeamento suave realizado pela rede, o que é característico de redes com poucos neurônios. A segunda vertente, considera o número de mapeamentos possíveis entre entrada e saída, o qual é descrito por Denker et al. (1987). Eles alertam que os dados de treinamento geralmente não impõem restrições às redes, e que normalmente existe um grande número de possíveis generalizações a partir de um dado conjunto de treinamento. Quanto maior o número de neurônios maior o número de possibilidades diferentes.

Blum (1992) & Masters (1993) sugerem que o número de neurônios ocultos num perceptron com apenas uma camada escondida deve estar relacionado com o número de unidades de entrada e o número de neurônios de saída, como pode ser visto a seguir,

$$Q_0 = \frac{M+P}{2} \quad \text{(regra do valor médio)} \tag{3.3}$$

$$Q_0 = 2P + 1 \quad \text{(regra de Kolmogorov)} \tag{3.4}$$

$$2\sqrt{P} + M \leq Q_0 \leq 2P + 1 \quad \text{(regra de Fletcher-Gloss)} \tag{3.5}$$

$$Q_0 = \sqrt{MP} \quad \text{(regra da raiz quadrada)} \tag{3.6}$$

onde Q_0 é o número de neurônios ocultos recomendado, M é o número de neurônios de saída e P é o número de unidades de entradas.

Essas regras simples têm pouco a oferecer além de um "chute" inicial para o número de neurônios ocultos. Elas ignoram o número de exemplos de treinamento, a quantidade de ruído nos dados e a complexidade da função de mapeamento. Mesmo quando se deseja minimizar o erro de treinamento com a imposição de que o conjunto de dados tem muitas amostras e nenhum ruído, é relativamente fácil construir contra-exemplos que desmentem essas regras.

Daqi & Yan (2005) consideram que o número de neurônios ocultos está apenas relacionado com o número de classes, assim como as regiões e as formas de distribuição das amostras no espaço de entrada, e não relacionados com o número de amostras ou dimensão do espaço de entrada. Porém, a determinação da distribuição das amostras em um espaço de alta dimensão é bastante difícil. Assim, eles apresentaram a seguinte fórmula empírica para selecionar o número inicial de neurônios ocultos:

$$Q_0 = [2log_2(P + K - 1)] \geq 2, \tag{3.7}$$

na qual K é o número de classes e P é o número de unidades de entrada. A Equação (3.7) é adequada para alguns problemas de classificação clássicos como XOR e IRIS.

Uma importante observação feita por Barron (1993), mediante a análise do erro quadrático médio de problemas de diversas magnitudes, é que para conjuntos de treinamento grandes, o erro para uma rede MLP (com uma camada escondida) é independente do tamanho do espaço de entrada e está relacionado com o inverso do número de neurônios ocultos ($O(1/Q)$). Então, MLPs são particularmente adequadas para lidar com problemas com grande dimensionalidade de entrada, ou seja, são menos susceptíveis à maldição da dimensionalidade que métodos clássicos de aproximação (e. g. polinômios).

Looney (1996) sugere algumas estratégias para determinar a configuração inicial da arquitetura da rede MLP em problemas de classificação. O número P de entradas é geralmente determinado pelo conjunto de treinamento. O número de neurônios de saída M pode ser igual ao número K de classes, ou se K é grande, pode ser $M = log_2(K)$. Para $M < log_2(K)$, o treinamento é mais difícil, mas possível. Isso deixa apenas o número de neurônios ocultos Q como o principal parâmetro arquitetural de projeto. Três formas de

proceder são listadas a seguir:

- usar o valor limite inferior de neurônios $Q = log_2(K)$ na camada escondida e adicionar mais neurônios até que a rede aprenda bem;
- No caso em que o número de subclasses[1] k atenda à condição $k \geq 2K$, iniciar a camada escondida com um número maior de neurônios ocultos do que seria necessário ($Q = [(k-1)/2 + (log_2(k))/2]$) para uma rede MLP com apenas uma camada escondida, treinar a rede, e então podar os neurônios ou pesos que não contribuam significantemente para o aprendizado (e retreinar após a poda);
- iniciar a rede MLP com duas camadas escondidas com tamanhos Q_1 e Q_2, em que $Q_1 > Q_2$ ($Q_1 = 2\sqrt{K}$ e $Q_2 = \sqrt{K}$), adicionar neurônios à primeira camada escondida conforme necessário, e finalmente podar a rede e retreinar.

A forma mais simples e, provavelmente a estratégia mais popular, para determinar a topologia da rede MLP é focar diretamente na minimização do erro quadrático médio sobre o conjunto de treinamento e avaliar a capacidade de generalização sobre um conjunto de teste. Entretanto, o excessivo esforço computacional geralmente é considerado uma grande dificuldade (CURRY; MORGAN, 2006).

3.3 Métodos Baseados em Inferência Estatística

Em projeto de MLPs, por qualquer que seja o método, efetivamente se está construindo um modelo não linear de um fenômeno físico responsável pela geração de exemplos do par entrada-saída a partir do treinamento da rede.

Pode-se supor que os vetores de entrada $\mathbf{x} \in X$ da rede MLP são independentes e distribuídos de acordo com uma função de distribuição de probabilidade $q(\mathbf{x})$ fixa, porém desconhecida. Pode-se assumir que o vetor de rótulos $\mathbf{y} \in Y$ é gerado por um operador desconhecido, o qual transforma os vetores $\mathbf{x}$ em vetores $\mathbf{y}$ de acordo com uma função de distribuição conjunta $q(\mathbf{x}, \mathbf{y}) = q(\mathbf{y}|\mathbf{x})q(\mathbf{x})$. Assim, a máquina de aprendizado observa os pares $(\mathbf{x}, \mathbf{y})$ e, durante o processo de treinamento, constrói alguma aproximação para o operador desconhecido, o qual pode ser usado para prever rótulos de acordo com a função

[1] As classes de uma população finita de vetores de características pode ser decomposta em subclasses convexas e linearmente separáveis. Estas subclasses podem ser separadas por um número apropriado de hiperplanos Q, que pode variar de um limite inferior $Q = log_2 k$ ($k = 2^Q$) a um limite superior $Q = k - 1$. A camada subsequente de neurônios junta as subclasses convexas em classes não convexas.

distribuição condicional $p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta})$, onde $\boldsymbol{\theta}$ é o vetor de parâmetros que especifica a rede, que são pesos e limiares.

Estimar esses parâmetros usando os dados é considerado ser a essência do problema de inferência estatística (MURATA et al., 1994; VAPNIK, 1998). Então, um grupo de redes neurais artificiais que têm a mesma arquitetura pode ser considerado como uma família paramétrica de distribuições condicionais e deve ser chamada de "modelo" e uma rede individual deve ser chamada de "máquina".

A questão é achar o modelo ótimo na família de redes e o conjunto de parâmetros ótimo para aproximar a distribuição condicional do sistema (MURATA et al., 1994). Formalmente, isso significa que em um subconjunto Θ do espaço vetorial $\Re^n$, um conjunto de funções admissíveis $G = \{g(\boldsymbol{\theta}), \boldsymbol{\theta} \in \Theta\}$ é dado, e um funcional, denominado funcional de risco, é definido como um critério de qualidade da função escolhida. Assim, o funcional ou função de custo $s(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta})$ a ser minimizada é

$$s(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}) \equiv \frac{1}{2} \int ||\mathbf{y} - g^{'}(\boldsymbol{\theta})||^2 p(g^{'}(\boldsymbol{\theta})|\mathbf{x}, \boldsymbol{\theta}) dg^{'}(\boldsymbol{\theta}) \tag{3.8}$$

onde $g'(\boldsymbol{\theta})$ é a predição feita pelo modelo.

É então requerido encontrar $g'(\boldsymbol{\theta})$ no conjunto G que minimiza o funcional e é suposto que o mínimo do funcional existe em G e corresponde ao modelo ótimo.

O cálculo do funcional de risco dado pela Equação (3.8) é bastante dificultado pelo desconhecimento de $p(g^{'}(\boldsymbol{\theta})|\mathbf{x}, \boldsymbol{\theta})$, e então, como em Engenharia o erro quadrático médio é a escolha natural como critério de qualidade para comparações entre modelos, o mesmo será adotado como o funcional de risco, só que agora, sob a denominação de funcional de risco empírico. Minimizá-lo é equivalente ao método dos mínimos quadrados (CURRY; MORGAN, 2006). Assim, o funcional de risco empírico ou função de custo $s(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta})$ a ser minimizada é

$$s(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}) \equiv \frac{1}{2N} \sum_{i=1}^{N} ||\mathbf{y}_i - f(\mathbf{x}_i, \boldsymbol{\theta})||^2, \tag{3.9}$$

onde $f(\mathbf{x}_i, \boldsymbol{\theta})$ representa a predição do modelo.

A questão agora, tratada pelo princípio da minimização do risco empírico, é trabalhar com o risco empírico dado pela Equação (3.9) e conhecer as condições sob as quais o mapeamento que minimiza o risco empírico está próximo do mapeamento desejado. Haykin (2002) afirma que quando o tamanho do conjunto de dados de treinamento cresce, o ponto mínimo do funcional de risco empírico converge em probabilidade para o ponto mínimo do funcional de risco verdadeiro. Então, é razoável avaliar o modelo neural pelo uso da

distribuição empírica (MURATA et al., 1994).

A taxa de convergência, tratada pela teoria da convergência uniforme do funcional de risco empírico para o funcional de risco real, tem limites, os quais são baseados na dimensão V-C (Vapnik-Chervonenkis). A estimativa da dimensão V-C é importante pois o número de exemplos necessários para se aprender de maneira confiável uma classe de interesse é proporcional à dimensão V-C daquela classe.

Embora, na maioria dos casos práticos, o cálculo analítico da dimensão V-C seja difícil de se fazer, alguns resultados sobre a dimensão V-C em redes neurais são conhecidos. Por exemplo, a dimensão V-C de um perceptron constituído por neurônios com funções de ativação de limiar é $O(WlogW)$, onde W é o número de parâmetros livres da rede. No caso de uma rede MLP cujos neurônios utilizam sigmóides como funções de ativação, a dimensão V-C é $O(W^2)$. Este resultado é particularmente importante pois apresenta um valor finito da dimensão V-C, e portanto, um tamanho finito para uma MLP (HAYKIN, 2002).

Outro candidado a função custo é o logaritmo da razão de verossimilhança

$$s(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}) \equiv ln \frac{q(\mathbf{y}|\mathbf{x})}{p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta})}. \tag{3.10}$$

Nesta função custo, a diferença entre a distribuição real, mas desconhecida, e a distribuição empírica é levada em consideração. Baseada nos dados de treinamento, a função custo pode ser apresentada como

$$s(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}) \equiv \frac{1}{N} \sum_{i=1}^{N} ln(q(\mathbf{y}_i|\mathbf{x}_i)) - \frac{1}{N} \sum_{i=1}^{N} ln(p(\mathbf{y}_i|\mathbf{x}_i, \boldsymbol{\theta})) \tag{3.11}$$

Para a identificação do modelo melhor ajustado, é necessário apenas a comparação dos valores das estimativas de $s(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta})$ para os vários modelos. Logo, não é necessário utilizar o primeiro termo do lado direito da Equação (3.11). Então, a função custo dada pela Equação (3.12) representa o negativo da média do logaritmo da verossimilhança (*mean log-likelihood* - MLL)

$$s(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}) \equiv -\frac{1}{N} \sum_{i=1}^{N} ln(f(\mathbf{x}_i, \boldsymbol{\theta})). \tag{3.12}$$

É importante notar que maximizar MLL significa aproximar a distribuição real pela distribuição empírica, e conseqüentemente, minimizar a função custo.

Observando a Equação (3.9), percebe-se que grandes erros de predição $(\mathbf{y}_i - f(\mathbf{x}_i, \boldsymbol{\theta}))$ dominam o valor final da função custo, pois suas contribuições são quadráticas. Já na Equação (3.12) este efeito é menor.

As Equações (3.9) e (3.12) são boas ferramentas para avaliar o grau de ajustamento do modelo ao conjunto de treinamento. Entretanto, é sabido que redes maiores atingem menores erros de treinamento, mas suas generalizações não são tão boas. Dessa forma, é necessário adotar algum critério que leve em consideração não apenas o ajustamento no treinamento, mas também a complexidade do modelo (i.e. número de parâmetros).

Neste sentido Moody (1992) introduz um termo de regularização para a função custo que penaliza a complexidade do modelo

$$d(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}) = s(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}) + r(\boldsymbol{\theta}), \tag{3.13}$$

onde o termo de penalização pode, por exemplo, ser dado por $r(\boldsymbol{\theta}) = ||\boldsymbol{\theta}||^2$.

Para regressão linear com o erro quadrático como função de custo existem algumas estimativas algébricas úteis para o desempenho esperado do estimador na predição de novas observações (risco de predição) que levam em consideração a complexidade do modelo. Duas das quais são apresentadas a seguir:

$$GCV = MSE \frac{1}{\left(1 - \frac{n_\theta}{N}\right)^2} \tag{3.14}$$

e

$$FPE = MSE \left(\frac{1 + \frac{n_\theta}{N}}{1 - \frac{n_\theta}{N}}\right) \tag{3.15}$$

onde MSE denota o erro quadrático médio no treinamento, n_θ o número de parâmetros ajustáveis do modelo e N o número de dados de treinamento.

A primeira, GCV, é a validação cruzada generalizada (CRAVEN; WAHBA, 1979; GOLUB et al., 1979) e a segunda, FPE, é o erro de predição final de Akaike (AKAIKE, 1970). A idéia é minimizar esses critérios.

Barron (1984) apresenta um critério para seleção de redes polinomiais chamado *Predicted Squared Error* (PSE):

$$PSE = MSE + 2\hat{\sigma}^2 \frac{n_\theta}{N}, \tag{3.16}$$

onde $\hat{\sigma}^2$ é uma estimativa prévia da variância do ruído nos dados, o qual não depende do modelo que se está considerando. A rede que atingir o menor PSE é escolhida como a

"melhor" (FOGEL, 1991).

PSE, FPE e GCV são estimativas que assintoticamente não apresentam viés (*unbiased*) em relação ao risco de predição para modelos neurais considerados aqui sob certas condições (MOODY, 1994), a saber: o ruído nos rótulos observados é independente e uniformemente distribuído, o modelo resultante não apresenta viés, decaimento de pesos não é usado e a não-linearidade no modelo pode ser negligenciada.

Para o PSE, ainda é necessário que seja usada uma estimativa de $\hat{\sigma}^2$ que assintoticamente não apresente viés. Na prática, contudo, essencialmente todos os ajustamentos realizados por redes neurais apresentam viés e/ou têm não-linearidades significantes.

Akaike (1974) propõe um critério de informação baseado numa função que pondera uma medida de ajustamento de um modelo MLL e o número de parâmetros independentes a serem ajustados para maximizar a verossimilhança. Esta estatística não requer julgamentos subjetivos. Uma modificação deste critério pode ser usada para a seleção da rede neural ótima.

Para combinar o critério da máxima verossimilhança e a escolha do modelo, pode-se penalizar a máxima verossimilhança (ou o seu logaritmo) com alguma função monotônica dependente do número de parâmetros livres relativos à complexidade do modelo. Tipicamente o critério tem a seguinte estrutura:

$$-2log(\hat{L}) + \alpha n_\theta \tag{3.17}$$

onde $\hat{L}$ é a verossimilhança estimada e n_θ é o número de parâmetros ajustáveis no modelo. A diferença básica entre os vários critérios, os quais podem ser derivados a partir de diferentes teorias, está na definição do termo de penalização da complexidade do modelo.

Com $\alpha = 2$, maximizar a verossimilhança penalizada significa minimizar AIC (AKAIKE, 1974):

$$AIC = -2log(\hat{L}) + 2n_\theta. \tag{3.18}$$

Nota-se que

$$e^{-\frac{1}{2}AIC} = \hat{L}.e^{-n_\theta}. \tag{3.19}$$

Assim a verossimilhança é modificada por um fator $exp(-n_\theta)$.

Com $\alpha = log(N) + 1$ tem-se o *Bozdogan's Consistent Akaike's Information Criterion* (HU; XU, 2004):

$$CAIC = -2log(\hat{L}) + (logN + 1)n_\theta \tag{3.20}$$

Com $\alpha = log(N)$ chega-se ao critério de informação bayesiana (*Bayesian Information Criterion*, BIC):

$$BIC = -2log(\hat{L}) + log(N)n_\theta. \tag{3.21}$$

Quando o tamanho da amostra é $N = 7389$, AIC e BIC coincidem em valor. O BIC tem se mostrado ser assintoticamente consistente para seleção de muitas classes de modelos, incluindo *mixture models* (KERIBIN, 1998), os quais compartilham muitas similaridades com redes neurais.

A análise baseada em AIC e BIC é uma aproximação para o ajustamento dos dados quando é levado em consideração que o modelo apresenta incerteza. É fácil de usar porque é simplesmente uma aproximação da máxima verossimilhança mais a correção do modelo.

Fogel (1991) modifica o critério de informação de Akaike (AIC) para aplicá-lo na seleção da "melhor" rede para problemas de classificação binária. A técnica pode ser estendida a problemas com um número arbitrário de classes. O critério de informação é chamado de *Final Information Statistic* (FIS):

$$FIS = -IQ + n_\theta, \tag{3.22}$$

onde IQ, quantidade de informação, leva em consideração os erros residuais de cada rótulo e as estimativas da máxima verossimilhança das distribuições das ativações do neurônio de saída para cada rótulo.

Deve-se ter cuidado ao usar o FIS. A estatística é baseada na consideração de que as distribuições das ativações são gaussianas. Esta consideração pode ser aproximada para redes grandes, mas pode não funcionar bem para redes pequenas. Além disso, IQ foi derivada a partir da consideração de que os padrões de entrada são estatisticamente independentes. Caso não sejam, a densidade conjunta não é mais simplesmente o produto de densidades individuais. Detalhes sobre implementação e dificuldades são discutidos em Fogel (1991).

Murata et al. (1994) generaliza o critério de informação de Akaike a partir do ponto de vista estatístico e propõe o chamado *Network Information Criteria* (NIC). O critério é dado por

$$NIC(f) = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(x_i, \theta_f))^2 + \frac{1}{N}traço(G_f^* Q_f^{*-1}). \tag{3.23}$$

O NIC assume que as funções de transferência são diferenciáveis, e assim G_f^* e Q_f^* são dados como a variância $V[d(s(x, y, \theta_f))/d\theta_f]$ e o valor esperado $E[d^2 s(x, y, \theta_f)/d\theta_f^2]$,

respectivamente, onde $s(x, y, \theta_f)$ é quase o mesmo que em (3.9), divergindo apenas pelo fator $1/2$.

Hagiwara et al. (2000) afirmam que o NIC se reduz a AIC quando a função de rótulo está na família mínima de redes. Em outras palavras, a rede que representa a verdadeira função é redundante.

Outro critério importante é o chamado *Minimum Description Length* (MDL) de Rissanen (1978). Este critério tem se mostrado consistente e é usado com sucesso na análise de modelos autoregressivos (AR) e autoregressivos de média móvel (ARMA) (GAO et al., 1997). Este princípio é baseado na teoria elementar de codificação (KENDALL et al., 1993). O comprimento descritivo (*description length*) de uma hipótese é a soma do número de bits necessários para codificar a hipótese e o número de bits necessários para codificar as exceções. Ele afirma que a melhor hipótese é aquela com menor comprimento descritivo. A Equação (3.21) formalmente coincide com o MDL(HU; XU, 2004).

te Brake et al. (1994) comparam os métodos MDL e NIC para a determinação do tamanho correto de uma rede MLP. Naturalmente, a maior diferença entre ambos está relacionado com seus princípios básicos, pois o MDL está relacionado com o comprimento do código e o NIC está relacionado com o erro médio esperado para novos exemplos. Outra grande diferença é que o NIC seleciona o modelo com a menor medida de erro, dependendo do tipo de erro escolhido. Já o princípio do MDL usa uma lista de exemplos classificados erroneamente e então, precisa de um limiar quando opera com números reais. Eles concluem que o NIC requer bastante esforço computacional se comparado ao MDL, pois precisa calcular matrizes inversas.

Lee (2001a) apresenta uma metodologia originada de aproximações bayesianas de escolha de modelos com a mais alta probabilidade *a posteriori*. Ele sugere que usando predições de apenas um modelo subestimará grosseiramente a variabilidade da estimativa, pois isto ignora o fato de que outro modelo com significante probabilidade *a posteriori* faça predição diferente. Ao invés, deve-se calcular predições pelo uso de uma média ponderada sobre todos os modelos no espaço de pesquisa, onde os pesos são as probabilidades *a posteriori* dos modelos (LEAMER, 1978; KASS; RAFTERY, 1995). Considerando y como a variável de resposta, D como os dados, e M_i como os modelos de interesse para $i \in L$, então a predição da distribuição *a posteriori* de y é dada por

$$P(y|D) = \sum_{i \in L} P(y|D, M_i) P(M_i|D), \tag{3.24}$$

onde $P(y|D, M_i)$ é a predição da densidade marginal *a posteriori* para um dado modelo

particular, e $P(M_i|D)$ é a probabilidade *a posteriori* do modelo i. Esta, embora conceitualmente clara, é um desafio computacional. Pelo Teorema de Bayes tem-se que

$$P(M_i|D) = \frac{P(D|M_i)P(M_i)}{\sum_j P(D|M_j)P(M_j)}, \tag{3.25}$$

onde $P(D|M_i) = \int f(y|p)f(p)$ é a probabilidade marginal dos dados, a qual é dada pelo produto da verossimilhança pela probabilidade *a priori* integrada sobre o espaço de parâmetros para o modelo. Esta integral é analiticamente intratável no caso de uma rede neural. Entretanto, a exponenciação do BIC dado pela Equação (3.21) oferece uma aproximação útil para a propabilidade *a posteriori* dos modelos, pois é uma aproximação para o logaritmo do fator Bayes($e^{-\frac{1}{2}BIC} = \hat{L}.(\sqrt{N})^{-n_\theta}$) (SCHWARZ, 1978). Na ausência de informações adicionais sobre o tamanho do modelo, pode-se adotar probabilidades iguais para cada modelo, isto é, $P(M_i) = P(M_j)$ para todos i e j. Assim, a aproximação para a probabilidade *a posteriori* do modelo i é dada por

$$P(M_i|D) = \frac{P(D|M_i)}{\sum_j P(D|M_j)} \approx \frac{e^{BIC_i}}{\sum_j e^{BIC_j}}. \tag{3.26}$$

Outros estudos sobre a aplicação de inferência estatística na seleção de modelos neurais podem ser encontrados nas seguintes referências (SEGHOUANE; AMARI, 2007; STOICA; SELéN, 2004; LEE, 2001b; BENGTSSON; CAVANAUGH, 2006; HSIEH, 2007; HU; XU, 2004; NAKAMURA et al., 2006; YUAN et al., 2003; ROSSI et al., 2006; MEDEIROS et al., 2006)

3.4 Métodos Construtivos

Alguns algoritmos iniciam o treinamento de uma rede MLP com apenas um neurônio oculto e adicionam novos neurônios sempre que a rede passa a ser incapaz de aprender novos padrões. Esses algoritmos têm obtido um sucesso razoável e podem oferecer outras vantagens, tal como redução de tempo de treinamento. Infelizmente, o desenvolvimento de um critério que interrompa a adição de novos neurônios sem lançar mão da validação cruzada ainda é um problema (KENDALL et al., 1993).

Moody (1994) propõe um algoritmo eficiente chamado *Sequential Network Construction* (SNC). O algoritmo constrói uma sequência de redes. Inicialmente, a rede com um número pequeno de neurônios ocultos é treinada, usando pesos iniciais aleatórios. Redes maiores são obtidas iterativamente pela adição de neurônios em blocos de tamanho C. Isso continua até que a rede atinja um tamanho máximo predeterminado. Quando uma rede maior é construída, esta é treinada como segue:

- Os pesos da rede prévia são usados como pesos iniciais para as primeiras unidades da nova rede. Os pesos dos neurônios do novo bloco C adicionado são iniciados com valores aleatórios pequenos;
- Os pesos do novo bloco de neurônios são treinados para um ótimo local com a manutenção dos demais pesos fixos;
- Finalmente, todos os pesos são liberados para adaptação e são treinados até que se atinja um ótimo local.

O algoritmo SNC possibilita alguma continuidade no espaço de modelos por construir uma sequência aninhada na qual uma rede maior contém redes menores. De fato isso significa que a medida que a sequência cresce, as redes são treinadas para aprender correções aos mapeamentos feitos anteriormente. Além disso, Moody alerta que quando a rede é grande o bastante, o efeito da adição de novos neurônios sobre os neurônios ocultos previamente existentes é pequeno.

Devido à resultante continuidade no espaço de modelos, pode-se ver o SNC como o inverso dos procedimentos de poda de rede.

A construção de grupos de modelos aninhados apresenta três vantagens. Primeira, a sequência terá um decréscimo (ou não crescimento) monotônico do erro de treinamento. Segunda, a sequência facilmente terá um mínimo identificável de risco de predição. Terceira, a seleção de arquitetura através de predição de risco tem uma conexão formal com estudos de teste de hipótese para poda quando o grupo de modelos é aninhado (MOODY, 1994).

Duas desvantagens de métodos de crescimento de estrutura são que a rede pode ser aprisionada num mínimo local e que eles são sensíveis às condições iniciais (MALDONADO; MANRY, 2002).

3.5 Métodos de Poda

Sietsma & Dow (1988) mostram que mesmo com um número apropriado de neurônios e com uma rede neural bem treinada, a mesma ainda pode ter benefícios com a redução de neurônios. Isto tende a reduzir o tamanho operacional, reduz o efeito do ruído e melhora a capacidade de generalização. Eles ainda sugerem heurísticas para identificar neurônios que não contribuem significativamente na composição do erro de saída da rede:

- Se o neurônio produz saída constante quando a rede é submetida ao conjunto de dados de entrada, então ele contribui apenas como um limiar (*bias*);
- Se dois neurônios têm saídas altamente correlacionadas, então um deles é redundante e pode ser removido.

Alguns pesquisadores têm voltado sua atenção para a aplicação de técnicas lineares, como *Principal Component Analysis* (PCA), para detectar baixa relevância de atributos ou neurônios ocultos na implementação de mapeamentos entrada-saída de redes neurais, mesmo estas sendo modelos não-lineares.

Outros têm trabalhado em técnicas de poda de pesos sinápticos individualmente. Mas, com a evolução do processo de poda, pode-se ter a completa eliminação de neurônios. Neste sentido, por exemplo, Corani & Guariso (2005) usam um método heurístico simples, no qual uma rede é inicialmente treinada e em seguida, alguns dos pesos com pequenas magnitudes (em valores absolutos) são eliminados. Neste método, algumas vezes chamado de método de poda pela força-bruta (REED, 1993), o pesquisador atribui zero ao peso em questão e avalia a mudança no erro cometido pela rede. Se ele cresce demais, então o valor do peso é restaurado. Caso contrário, o peso é zerado permanentemente. Entretanto, a simples magnitude do peso não é um parâmetro suficiente para avaliar o efeito deste peso sobre o erro de saída da rede (LEVIN; MOODY, 1994). Uma forma mais criteriosa de poda da rede é proposta no método *Skeletonization* (MOZER; SMOLENSKY, 1989). Neste método a mudança no erro é avaliada em relação a retirada de todos os pesos, um de cada vez, e então, aquele que produz o menor efeito no erro é zerado. Esta operação é repetida enquanto o erro cometido pela rede estiver dentro de um limiar previamente estabelecido.

Muitos dos algoritmos tratados nesta subseção podem ser agrupados em dois grupos gerais. No primeiro, os algoritmos estimam a sensibilidade da função de erro em relação à remoção dos pesos. Os pesos com o menor efeito é removido. No segundo grupo, é adicionado um termo à função de custo com o objetivo de tornar a solução da rede mais eficiente. Por exemplo, um termo proporcional à soma dos valores absolutos ou quadráticos de todos os pesos favorece a soluções com pesos pequenos. Aqueles pesos com valores próximos a zero podem não ter muita influência sobre a saída e então podem ser eliminados. Existe uma certa sobreposição entre estes dois grupos, pois a função custo pode conter termos de sensibilidade.

3.5.1 Métodos Relacionados com PCA

Chen et al. (2001) sugerem que o número de neurônios ocultos pode ser reduzido pela identificação de neurônios linearmente redundantes, isto é, as saídas dos quais podem ser representadas por uma combinação linear das saídas dos outros neurônios na mesma camada. Os métodos apresentados a seguir são baseados de alguma forma neste princípio.

Levin & Moody (1994) propõem o método *Principal Components Pruning* (PCP). Ele é baseado na análise de componentes principais das ativações dos neurônios de camadas sucessivas da rede. São requeridos apenas os pesos e as matrizes de correlação das ativações dos neurônios para cada camada.

Considerando uma MLP, na qual a saída da camada i é genericamente representada da forma

$$\mathbf{y}_i = \Gamma[\mathbf{W}_i \mathbf{x}_i] \equiv \Gamma[\mathbf{u}_i]. \tag{3.27}$$

em que, $\mathbf{x}_i$ é o vetor de entrada da camada, $\mathbf{u}_i$ é o vetor das ativações da camada, Γ é um operador consistindo de funções de ativação dos neurônios na camada, e $\mathbf{y}_i$ é o vetor de saída da camada i, o procedimento é:

Passo 1: Treinar a rede usando o procedimento de treinamento por retropropagação;

Passo 2: Partindo da primeira camada, a matriz de correlação Σ entre os vetores de entrada e os respectivos vetores de saída da primeira camada deve ser calculada;

Passo 3: As componentes principais devem ser ordenadas pelos seus efeitos na saída linear da camada;

Passo 4: Montar uma matriz $\mathbf{C}$, cujas colunas sejam os autovetores da matriz de correlação e avaliar o efeito da remoção de cada autovetor usando um conjunto de validação. Aqueles autovetores cuja remoção não produz aumento no erro de validação devem ser eliminados;

Passo 5: Os pesos da camada devem ser projetados no subespaço dimensional l expandido pelos l autovetores significantes

$$\mathbf{W} \rightarrow \mathbf{W}\mathbf{C}_l\mathbf{C}_l^T; \tag{3.28}$$

Passo 6: O procedimento deve continuar até que todas as camadas sejam podadas.

O efeito do método PCP é reduzir o posto de cada camada de pesos numa rede pela

remoção dos autovetores menos significantes (MOODY, 1992). É importante observar que a aplicação deste procedimento não reduz o esforço computacional, pois nenhum neurônio foi realmente eliminado. Entretanto, em geral, a capacidade de generalização da rede melhora tal como acontece por ocasião da aplicação de métodos de poda. O próximo procedimento efetivamente reduz a topologia da rede.

Chen et al. (2001) propõem outro método de poda baseado em PCA. O procedimento é apresentado a seguir:

Passo 1: Denotar $\mathbf{x}(t) = [x_1(t)\ x_2(t)\ ...\ x_k(t)]$ como o vetor que representa as saídas dos k neurônios da i-ésima camada, onde t é o índice da amostra;

Passo 2: Normalizar as saídas de cada neurônio com média 0 e variância 1 e montar uma matriz $\mathbf{X}_s$ com as saídas $\mathbf{x}(t)$ normalizadas em cada linha;

Passo 3: Calcular as componentes principais de $\mathbf{X}_s$. Estas são as combinações lineares $\mathbf{t}_i = \mathbf{p}_i^T \mathbf{X}_s$, onde $||\mathbf{p}_i|| = 1$ é o i-ésimo autovetor da matriz de covariância de $\mathbf{X}_s$. As componentes principais ($\mathbf{t}_i$), ditas variáveis latentes, são ordenadas decrescentemente pelos valores de seus autovalores(λ_i);

Passo 4: Escolher l componentes principais de tal forma que

$$\sum_{i=1}^{l} \lambda_i \geq \eta \sum_{i=1}^{k} \lambda_i, \tag{3.29}$$

onde η tipicamente está na faixa entre $0,8$ e $0,85$.

Passo 5: Para cada $\mathbf{t}_i$ $(1 \leq i \leq l)$, determinar um neurônio cuja saída seja mais correlacionada com $\mathbf{t}_i$;

Passo 6: Remover os $k - l$ neurônios restantes na i-ésima camada;

Passo 7: Se não existem mais neurônios a serem podados, fim. Senão, treinar a rede podada e ir para o passo 1.

Diferentemente do método de Chen et al. (2001), onde são consideradas apenas as relações entre os neurônios na mesma camada, Henrique et al. (2000) propõem o método de poda *Orthogonal Least Squared*(OLS). Eles eliminam os neurônios que são insignificantes para a próxima camada. Entretanto, o método proposto só pode ser aplicado a situações em que a $(i+1)$-ésima camada tem apenas um neurônio.

O método de poda OLS é apresentado como segue:

Passo 1: Denotar $\mathbf{x}(t) = [x_1(t)\ x_2(t)\ ...\ x_k(t)]$ como as saídas dos neurônios da i-ésima camada, $y(t)$ como a entrada do nó da $i+1$-ésima camada, em que t é o número da amostra;

Passo 2: Arranjar $[\mathbf{x}(1), \mathbf{x}(2), ..., \mathbf{x}(n)]$ numa matriz de dados $\mathbf{X}_{n\mathbf{x}k}$ e $[y(1), y(2), ..., y(n)]$ num vetor de dados $\mathbf{y}$ (n x 1), em que n é o número total de amostras e k é o número de neurônios da i-ésima camada. Tem-se que $\mathbf{y} = \mathbf{Xw}$, onde $\mathbf{w}$ (k x 1) é o vetor de pesos;

Passo 3: Considerar a fatorização de Choleski de $\mathbf{X}$,

$$\mathbf{X^T X} = \mathbf{A^T D A}, \tag{3.30}$$

onde $\mathbf{A}$ é uma matriz triangular superior com elementos unitários na diagonal principal e $\mathbf{D}$ é uma matriz com elementos positivos na diagonal principal. A relação entre $\mathbf{X}$ e $\mathbf{y}$ pode ser modificada para

$$\mathbf{y} = \mathbf{Bg}, \tag{3.31}$$

onde

$$\mathbf{B} = \mathbf{XA}^{-1} = [\mathbf{b}_1\ \mathbf{b}_2\ ...\ \mathbf{b}_k] \tag{3.32}$$

e $\mathbf{g} = \mathbf{Aw} = [g_1\ g_2\ ...\ g_k]^T$. $\mathbf{B}$ é também chamada matriz regressora auxiliar e possui colunas ortogonais pois $\mathbf{B}^T\mathbf{B} = (\mathbf{XA}^{-1})^T(\mathbf{XA}^{-1}) = \mathbf{D}$;

Passo 4: Agora que $\mathbf{B}$ é ortogonal, $\mathbf{y}^T\mathbf{y}$ é dado por

$$\mathbf{y}^T\mathbf{y} = \sum_{i=1}^{k}[g_i^2\mathbf{b}_i^T\mathbf{b}_i]. \tag{3.33}$$

Entretanto, a contribuição do i-ésimo neurônio para a variância de y que é explicada pelo regressor auxiliar $\mathbf{b}_i$ é dada por $g_i^2\mathbf{b}_i^T\mathbf{b}_i$. Este resultado pode ser usado como critério para seleção da estrutura da rede neural;

Passo 5: Remover os neurônios com baixo valor de $g_i^2\mathbf{b}_i^T\mathbf{b}_i$ (valor de saliência);

Passo 6: Se não existem mais nós para serem podados, fim. Senão, treinar a rede podada e ir para o passo 1.

Comparado com o método PCA, o método OLS considera mais a relação entre neurônios de camadas sucessivas do que nós da mesma camada. Logo, estes dois métodos são complementares.

3.5.2 Métodos Baseados em Termos de Penalização

Os métodos baseados em termos de penalização modificam a função custo de tal forma que a retropropagação baseada nessa função leva os pesos desnecessários a zero e, na realidade, os remove automaticamente durante o treinamento. Mesmo quando os pesos não são realmente removidos, a rede se comporta como um sistema menor (REED, 1993).

Chauvin (1989) usa a função custo

$$C = \mu_{oe} \sum_{t=1}^{N} \sum_{k=1}^{M} \left[d_k(t) - y_k^{(o)}(t) \right]^2 + \mu_{he} \sum_{t=1}^{N} \sum_{i=1}^{Q} \nu(\xi_i(t)) \tag{3.34}$$

em que ν é uma função monótona positiva da "energia" dispendida por uma unidade de processamento. Os somatórios ocorrem sobre todo o conjunto de dados levando em consideração os neurônios de saída M, os neurônios ocultos Q e os N padrões. O primeiro termo é o termo de erro normalmente utilizado na retropropagação, mas o segundo termo mede a "energia" média (ξ) dispendida pelos neurônios ocultos. Os parâmetros μ_{oe} e μ_{he} balanceiam o efeito dos dois termos.

A energia dispendida por uma unidade ($\xi_i = (y_i^{(h)})^2$) - o quanto sua atividade varia em função dos padrões de treinamento - é uma indicação de sua importância. Se a saída de uma unidade muda bastante, provavelmente esta unidade proporciona informação significante. Se não muda muito, provavelmente não fornece muita informação.

Comportamentos qualitativamente diferentes podem ser vistos dependendo da forma da função ν (REED, 1993). Várias funções foram examinadas cujas derivadas são

$$\nu' = \frac{d\nu(\xi)}{d\xi_i} = \frac{1}{(1+\xi)^n}, \tag{3.35}$$

em que n é um inteiro. Para $n = 0$, ν é linear, e assim energias altas e baixas recebem penalidades iguais. Para $n = 1$, ν é logaritmica, e as unidades com baixa energia são penalizadas mais que aquelas com alta energia. Para $n = 2$, a penalização tem comportamento assintótico com o aumento da energia, de tal forma que unidades com alta energia não são muito mais penalizadas que as unidades com média energia.

Um termo que leva em conta a magnitude dos pesos pode ser adicionado à função custo, resultando em

$$C = \mu_{oe} \sum_{t=1}^{N} \sum_{k=1}^{M} \left[d_k(t) - y_k^{(o)}(t) \right]^2 + \mu_{he} \sum_{t=1}^{N} \sum_{i=1}^{Q} e(\xi_i(t)^2) + \mu_\theta g(\boldsymbol{\theta}). \tag{3.36}$$

Os fatores de ponderação (μ_{oe}, μ_{he} e μ_θ) na Equação (3.36) requerem algum ajuste e dependem do problema. Mas o que é mais comum é a desconsideração do termo referente à energia $\mu_{he} = 0$ e o ajuste de μ_{oe} e μ_θ manualmente por testes de desempenho na generalização sobre um conjunto de dados independente.

Ishikawa (1990) propõe a Equação (3.37) como função de pesos. A derivada de $g(\boldsymbol{\theta})$ em função de θ_i adiciona $-\mu_\theta sgn(\theta_i)$ à regra de atualização de pesos[2]. Se $\theta_i \geq 0$, o peso é decrementado de μ_θ, caso contrário, é incrementado de μ_θ.

$$g(\boldsymbol{\theta}) = \sum_i^W |\theta_i| \tag{3.37}$$

Hinton (1989) usa o método de decaimento de pesos, no qual o termo de penalização é dado por

$$g(\boldsymbol{\theta}) = \sum_i^W \theta_i^2 = ||\boldsymbol{\theta}||^2. \tag{3.38}$$

Dessa forma, considerando que a derivada de $g(\boldsymbol{\theta})$ em função de θ_i é $2\mu_\theta\theta_i$, isso efetivamente introduz um termo de decaimento de pesos nas equações de retropropagação. Pesos maiores são mais penalizados durante o processo de otimização e pesos que não são essenciais para a solução decaem a zero e podem ser removidos.

A desvantagem do termo de penalização $\sum_i \theta_i^2$ é que ele tende a favorecer vetores de pesos com muitos componentes pequenos ao invés daqueles com apenas um componente grande, mesmo quando isto é desejado (LOONEY, 1996).

Um método similar chamado eliminação de pesos é proposto por Weigend et al. (1991). O termo de penalização é dado por

$$g(\boldsymbol{\theta}) = \sum_{i \in C_{total}} \frac{(\theta_i/\theta_0)^2}{1 + (\theta_i/\theta_0)^2}, \tag{3.39}$$

em que θ_0 é um parâmetro livre predefinido do procedimento de eliminação de pesos e o conjunto C_{total} se refere a todas as conexões sinápticas na rede.

Kendall et al. (1993) derivam uma função custo para decaimento de pesos sob a perspectiva do critério MDL. Eles consideraram tal critério como o negativo do logaritmo da probabilidade dos parâmetros e assumiram que tanto a distribuição do erro de saída da rede quanto a distribuição dos pesos seriam modelados por funções gaussianas simples com média zero e com iguais variâncias. Eles consideraram que o o comprimento do código

[2] A função sinal ($sgn(\theta_i)$) é uma função de limiar que assume valor igual a 1 se $\theta_i \geq 0$, e valor igual a -1 se $\theta_i < 0$.

de cada entrada é o mesmo que o de cada saída, e então μ_θ pode ser ajustado puramente na base do raciocínio de entradas e saídas. Isso dá

$$\mu_\theta = \frac{M}{M+P} \tag{3.40}$$

em que M é o número de saídas da rede e P é o número de entradas da rede.

Mackay (1992) introduz uma técnica para ajustar μ_θ automaticamente durante o treinamento baseado na perspectiva baysiana.

3.5.3 Métodos Baseados no Cálculo de Sensibilidades

A idéia fundamental dos algoritmos de poda baseados em cálculos de sensibilidades é partir de uma rede completamente conectada e com o número de parâmetros supostamente suficiente para aprender a relação entre entrada e saída, calcular alguma medida de contribuição (sensibilidade) de cada peso para a solução do problema, e então podar aqueles pesos com menores influências na rede, resultando em uma rede com menos pesos e parcialmente conectada.

LeCun et al. (1990) propõem um modelo local da superfície de erro para prever analíticamente o efeito da retirada de cada peso sobre o erro de saída da rede (E_{med}). A expansão em série de Taylor foi utilizada para aproximar o erro quadrático médio como pode ser visto na expressão a seguir

$$E_{med}(\bar{\boldsymbol{\theta}} + \Delta\boldsymbol{\theta}) = E_{med}(\bar{\boldsymbol{\theta}}) + g^T(\bar{\boldsymbol{\theta}})\Delta\boldsymbol{\theta} + \frac{1}{2}\Delta\boldsymbol{\theta}^T\mathbf{H}(\bar{\boldsymbol{\theta}})\Delta\boldsymbol{\theta} + O(||\Delta\boldsymbol{\theta}||^2), \tag{3.41}$$

onde $\Delta\boldsymbol{\theta}$ é a perturbação, em torno do ponto de operação ($\bar{\boldsymbol{\theta}}$), no vetor de pesos devido a retirada de um peso específico. $g^T(\bar{\boldsymbol{\theta}})$ e $\mathbf{H}(\bar{\boldsymbol{\theta}})$ são respectivamente o vetor gradiente e a matriz Hessiana ($\mathbf{H}(\bar{\boldsymbol{\theta}}) = d^2E_{med}(\bar{\boldsymbol{\theta}})/d\boldsymbol{\theta}^2$) calculados no ponto de operação ($\bar{\boldsymbol{\theta}}$).

Considerando que o treinamento exaustivo com convergência leva a rede a um mínimo global ou local e que a superfície de erro em torno deste mínimo é aproximadamente quadrática, a Equação (3.41) pode ser simplificada como segue

$$E_{med}(\bar{\boldsymbol{\theta}} + \Delta\boldsymbol{\theta}) = E_{med}(\bar{\boldsymbol{\theta}}) + \frac{1}{2}\Delta\boldsymbol{\theta}^T\mathbf{H}(\bar{\boldsymbol{\theta}})\Delta\boldsymbol{\theta} \tag{3.42}$$

e a variação no erro de saída em função da remoção de um peso pode ser dada por

$$\Delta E_{med} = \Delta\boldsymbol{\theta}^T\frac{\mathbf{H}(\bar{\boldsymbol{\theta}})}{2}\Delta\boldsymbol{\theta}. \tag{3.43}$$

A idéia é minimizar a Equação (3.43) sujeito a condição $\Delta\theta_i + \theta_i = 0$, ou seja, sujeito a remoção do peso θ_i. Isto recai num problema de otimização com restrições cuja solução é obtida a partir do *lagrangiano*

$$S = \Delta\boldsymbol{\theta}^T \frac{\mathbf{H}(\bar{\boldsymbol{\theta}})}{2} \Delta\boldsymbol{\theta} - \lambda(\mathbf{1}_i^T \Delta\boldsymbol{\theta} + \theta_i) \tag{3.44}$$

onde λ é o *multiplicador de Lagrange* e $\mathbf{1}_i$ é um vetor unitário cujos elementos são todos zeros, exceto o i-ésimo elemento. A solução deste problema fornece uma expressão que reflete a variação do erro devido a remoção do peso θ_i, Equação (3.45), bem como a modificação ótima do vetor de peso $\Delta\boldsymbol{\theta}$, Equação (3.46), mostradas a seguir

$$S_i = \Delta E_{med}(i) = \frac{1}{2} \frac{\theta_i^2}{[\mathbf{H}(\bar{\boldsymbol{\theta}})^{-1}]_{i,i}} \tag{3.45}$$

$$\Delta\boldsymbol{\theta} = \frac{\theta_i^2}{[\mathbf{H}(\bar{\boldsymbol{\theta}})^{-1}]_{i,i}} \mathbf{H}(\bar{\boldsymbol{\theta}})^{-1} \mathbf{1}_i \tag{3.46}$$

onde $[\mathbf{H}(\bar{\boldsymbol{\theta}})^{-1}]_{i,i}$ é o elemento (i,i) da matriz inversa de $\mathbf{H}(\bar{\boldsymbol{\theta}})$. S_i é conhecida como a saliência ou sensibilidade do peso θ_i.

LeCun et al. (1990) simplifica o cálculo das sensibilidades fazendo a imposição de que a Hessiana seja uma matriz diagonal e intitula seu método como *Optimal Brain Damage* (OBD). Esta simplificação reduz substancialmente o esforço computacional. Entretanto, Hassibi et al. (1993) demonstra que a computação completa da matriz Hessiana torna o método mais poderoso e o intitula *Optimal Brain Surgeon* (OBS).

O processo de aplicação do OBS pode ser automatizado como segue:

Passo 1: Treinamento de uma arquitetura completamente conectada com parâmetros o bastante para capturar a relação entrada-saída;

Passo 2: Ordenar os parâmetros com base em suas sensibilidades;

Passo 3: Eliminação do peso com a menor sensibilidade e geração de uma nova arquitetura;

Passo 4: Embora a Equação (3.46) seja utilizada para atualização do vetor de pesos da rede obtida, é recomendável retreinar a rede sempre que um peso ou uma pequena parte deles (3-5%) é eliminado (HASSIBI et al., 1993; HAYKIN, 2002; CORANI; GUARISO, 2005);

Passo 5: Avaliação de seu desempenho nos conjuntos de treinamento e teste;

Passo 6: Retornar ao Passo 2 enquanto a poda produzir resultados aceitáveis segundo algum critério previamente estabelecido.

No caso do OBD, recomenda-se que apenas um peso deve ser eliminado por vez em virtude da simplificação da Hessiana. Dessa forma, a sensibilidade S_i do i-ésimo peso é definida como

$$S_i = \frac{1}{2}\frac{d^2 E_{med}}{d\theta_i^2}.\theta_i^2 \tag{3.47}$$

Karnin (1990) propõe uma medida aproximada de sensibilidade que pode ser usada durante o processo de treinamento. Ao invés de realmente remover o peso e calcular o erro com a sua remoção, a Equação (3.48) aproxima a sensibilidade pelo monitoramento da soma de todas as mudanças nos pesos durante o treinamento:

$$\hat{S}_i = -\sum_{n=0}^{N-1} \frac{dE}{d\theta_i}\Delta\theta_i(n)\frac{\theta_i^F}{\theta_i^F - \theta_i^I} \tag{3.48}$$

onde N é o número de épocas de treinamento, θ_i^I é o peso inicial, θ_i^F é o peso final e $\Delta\theta_i(n)$ é a atualização do peso θ_i na época n. Todos estes termos são disponíveis durante o treinamento, e assim, a expressão é fácil de calcular e não há uma fase específica para cálculo de sensibilidades. Quando $\Delta\theta_i$ é calculado pela retropropagação sem termo de momento, a Equação 3.48 passa a ser escrita como

$$\hat{S}_i = \sum_{n=0}^{N-1} [\Delta\theta_i(n)]^2 \frac{\theta_i^F}{\theta_i^F - \theta_i^I}. \tag{3.49}$$

Após o treinamento, cada peso tem uma sensibilidade estimada e o peso com a mais baixa sensibilidade pode ser eliminado.

Finnoff et al. (1993) propõem outro método para calcular o coeficiente de importância (T) do peso que pode ser calculado a qualquer instante durante o treinamento. Em contraste com o OBS e OBD, esta medida não assume que um mínimo na superfície de erro tenha sido atingido. Na realidade T é uma estatística de teste para a hipótese de que um peso venha a ser zero durante o processo de treinamento:

$$T(\theta_i) = log\left(\frac{|\sum_N \theta_i - \eta(dE/d\theta_i)_N|}{\eta\sqrt{\sum_N((dE/d\theta_i)_N - \overline{(dE/d\theta_i)})^2}}\right) \tag{3.50}$$

Na equação acima, os somatórios são sobre todos os N exemplos do conjunto de treinamento, η é a taxa de aprendizado, e o elemento barrado indica média amostral

sobre os exemplos. Um valor grande de T indica alta importância do peso θ_i. Conexões com pequenos valores de T podem ser podadas.

Os autores sugerem podar quando começa haver sobreajustamento dos dados, por exemplo, quando o erro no conjunto de validação dobra durante o treinamento. Sobre o número de parâmetros a serem removidos eles sugerem remover 35% dos pesos na primeira poda e 10% em cada passo posterior. Tais heurísticas, entretanto, não são satisfatórias, pois obviamente não podem ser ótimas sempre. Assim, a seleção do número de pesos a serem removidos em cada passo de poda é uma questão importante.

Prechelt (1995) propõe um método de poda chamado *Lprune* que adapta automaticamente a extensão da poda à evolução dos pesos e à perda de generalização durante o treinamento. O método é baseado na distribuição da estatística T durante o treinamento e não requer ajuste de parâmetros de algorítmo pelo usuário. O número de pesos a serem podados é uma fração (λ) da média (μ_T) de todos os valores $T(\theta_i)$. Dessa forma, a regra é: em cada passo de poda, podar todas aquelas conexões i cujos pesos θ_i satisfazem $T(\theta_i) < \lambda\mu_T$ para algum $\lambda \in [0, 1]$.

O fator λ é ajustado dinamicamente por

$$\lambda = \lambda(GL) = \lambda_{max}\left(1 - \frac{1}{1 + \frac{GL}{\alpha}}\right) \tag{3.51}$$

onde $\lambda_{max} = 2/3$ e $\alpha = 2$ foram encontrados por experimentação. Estes parâmetros são moderadamente críticos. GL é uma medida de sobreajustamento dos dados dada por

$$GL = GL(t) = 100.\left(\frac{E_{va}(t)}{E_{opt}(t)}\right) \tag{3.52}$$

cujo $E_{va}(t)$ é o erro quadrático no conjunto de validação na época t e $E_{opt}(t)$ é o erro quadrático mais baixo no conjunto de validação obtido até a época t.

É importante observar que para um alto grau de sobreajustamento de dados, mais pesos serão podados. Prechelt (1995) conclui por experimentação que a adaptação automática da extensão da poda pode resultar em redes melhores que aquelas obtidas com esquemas de poda estáticos, principalmente para redes pequenas. Entretanto, o *Lprune* é apenas parcial, pois é inábil para executar podas severas em estágios iniciais do treinamento como as vezes é necessário, em particular para redes com demasiado número de entradas, tais como aquelas usadas em problemas de bioinformática.

Em Moody & Utans (1992) é proposto um método simples de poda de variáveis de entrada x_i baseado em sensibilidade. A sensibilidade do modelo neural para a variável i

é definida como:

$$S_i = \frac{1}{N} \sum_t S_i(t) \tag{3.53}$$

em que $S_i(t)$ é a sensibilidade calculada para o padrão de entrada $\mathbf{x}(t)$. Considerando que geralmente existem muito menos entradas que pesos, a avaliação direta de S_i é factível:

$$S_i(t) = \left(\frac{1}{N} \sum_{t=1}^{N} x_i(t) \right) - x_i(t) = \bar{x}_i - x_i(t). \tag{3.54}$$

S_i é utilizada como uma estimativa de quanto o erro quadrático no conjunto de treinamento é afetado pela substituição do i-ésimo atributo x_i por sua média $\bar{x}_i$ em todos as amostras. A substituição de uma variável de entrada por sua média remove sua influência sobre a saída da rede.

3.5.4 Outros Métodos

Hayashi (1993) sugere que o posto da matriz de covariância ($Q \times P$), obtida pelas Q ativações da camada escondida sob a aplicação de N vetores de entrada, é igual ao número de unidades ocultas necessárias à rede. Bons resultados foram apresentados em problemas de reconhecimento de caracteres.

Teoh et al. (2006) quantificam a significância do acréscimo no número de neurônios na camada oculta de uma rede MLP usando a decomposição em valores singulares (SVD). Eles medem a capacidade de generalização de uma rede MLP com apenas uma camada oculta através do grau de independência linear dos padrões projetados no espaço de saída da camada oculta, o qual pode ser indiretamente quantificado a partir dos valores singulares obtidos através da SVD.

Yacoub & Bennani (2000) propõem um método chamado *Heuristic for Variable Selection* (HVS), no qual as variáveis importantes no espaço de características são identificadas e selecionadas. Aquelas que são redundantes ou não contêm informação relevante o bastante são eliminadas. Este método pode ser estendido para otimização de arquitetura. A saída de cada camada escondida pode ser vista como uma camada de entrada para uma sub-rede formada por esta camada, a camada de saída e todas as camadas entre estas duas camadas. Assim, o problema de otimização da arquitetura vem a ser um problema de seleção de variáveis numa sub-rede. A contribuição parcial da unidade j conectada à

unidade i através do peso θ_{ij} é definida como

$$\pi_{ij} = \frac{|\theta_{ij}|}{\sum_{k \in fan-in(i)} |\theta_{ik}|}, \tag{3.55}$$

em que *fan-in(i)* é o conjunto de unidades que têm conexão com a unidade i. A contribuição relativa da unidade j para a saída de decisão da rede é a soma ponderada de suas contribuições parciais pelas contribuições relativas das unidades para as quais são enviadas conexões, como pode ser visto em

$$S_j = \sum_{k \in fan-in(i)} \pi_{ij} \delta_i, \tag{3.56}$$

em que δ_i assume valor igual a 1 se a unidade i pertence à camada de saída, e S_i se a unidade i pertence à camada oculta.

O algoritmo HVS para seleção de variáveis é o seguinte:

Passo 1: Treinar a rede usando o procedimento de parada prematura (*early stopping*);

Passo 2: Calcular a pertinência de cada variável de entrada de acordo com a medida HVS;

Passo 3: Podar a variável de entrada com menor pertinência;

Passo 4: Retornar ao passo 1 até a última variável.

O algoritmo gera um conjunto de redes com cada vez menos variáveis ou neurônios. Assim, deve-se escolher a rede que obtenha o menor erro no conjunto de validação.

3.6 Conclusões

Este capítulo tratou da seleção da topologia de MLPs com uma única camada escondida. Critérios e metodologias oriundas de diversas teorias foram apresentadas como ferramentas para a escolha ou determinação de topologias que associem características desejáveis em um modelo neural tais como representatividade do mapeamento entrada-saída, capacidade de generalização e baixo custo computacional.

Inicialmente foram apresentadas algumas regras heurísticas, as quais, em sua maioria, são frutos de extensiva experimentação com diversos problemas e se apresentam como recomendações de cunho geral. Apresentam fortes limitações principalmente por não

levarem em conta a complexidade do problema em questão. Foram também abordados alguns critérios, baseados em inferência estatística, que consideram a complexidade do problema. Estes critérios estabelecem índices que servem como base de comparação entre modelos pertencentes a famílias de redes neurais. A questão está relacionada com a relação entre a representatividade do modelo e a complexidade da topologia.

Por fim, foram apresentadas algumas metodologias para redução de complexidade de topologias previamente definidas. Algumas são baseadas na teoria da regularização, onde há a inserção de termos de penalização de complexidade na função custo. Neste caso, a eliminação de pesos é realizada durante o processo de treinamento. Outras metodologias realizam a redução da estrutura pela poda de pesos e/ou neurônios de redes MLP previamente treinadas. A poda é baseada na avaliação da sensibilidade da função custo em relação a eliminação de cada peso individualmente.

Os métodos baseados na poda de redes MLP previamente treinadas são bastante atrativos, pois mesmo que uma MLP totalmente conectada apresente bons resultados, a poda de pesos ou neurônios redundantes ainda pode melhorar a capacidade de generalização da rede e reduzir a complexidade do modelo. Porém, o cálculo de medidas de sensibilidade pode requerer algoritmos sofisticados.

O capítulo a seguir é dedicado à apresentação de uma nova metodologia de poda de pesos de redes MLP previamente treinadas baseada na medida da correlação entre erros. Os cálculos envolvidos nesta metodologia apresentam baixa complexidade.

4 *Metodologia Proposta para Poda de Conexões Sinápticas em MLPs*

Conforme discutido no capítulo 2, redes MLP projetadas para funcionar com elevado número de parâmetros geralmente aprendem relativamente rápido e apresentam menor sensibilidade à inicialização dos pesos. Entretanto, topologias de maior porte apresentam alguns incovenientes quando se pretende implementá-las, por exemplo, em sistemas embarcados, tal como grande demanda de memória e, no caso de aplicações em tempo real, o tempo de execução da rede pode ser inaceitável. Além disso, topologias com parâmetros em excesso são mais propensas à ocorrência de *overfitting*. A aplicação de um método de poda de pesos leva quase sempre a uma redução da estrutura ao eliminar pesos desnecessários (ou redundantes), o que diminui a demanda por memória e o tempo de processamento, além de geralmente melhorar a capacidade de generalização da rede.

Neste capítulo é apresentada uma metodologia eficiente de poda de conexões sinápticas excedentes em uma rede MLP previamente treinada. O princípio subjacente à metodologia proposta é chamado de *Princípio da Máxima Correlação dos Erros* (MAXCORE), pois é baseado na análise das matrizes de correlação dos erros produzidos pelos neurônios de uma dada camada e os erros retropropagados para os neurônios da camada antecessora, não requerendo assim inversões de matrizes hessianas ou ajuste de quaisquer parâmetros de regularização.

O princípio MAXCORE dá margem ao surgimento de um algoritmo de poda, que recebe o nome de CAPE (**C***orrelation* **A***nalysis of back-***P***ropagated* **E***rrors*), cuja aplicação sucessiva pode levar, eventualmente, a uma completa eliminação de todas as conexões de certos neurônios da rede.

4.1 O Princípio da Máxima Correlação dos Erros

Conforme apresentado na seção 2.1 o treinamento de uma rede MLP é realizado em duas etapas básicas. Na primeira, quando cada padrão é apresentado à rede, o fluxo de informação (fluxo direto) segue da camada de entrada para a camada de saída. A relação entrada-saída é não-linear, pois os neurônios são equipados com funções de ativação sigmoidais, geralmente funções logísticas ou tangentes hiperbólicas. Após o cálculo do erro de saída se dá a segunda etapa. Os pesos são atualizados, camada a camada, em função dos erros retropropagados a partir do erro na camada de saída. Aqui, o fluxo de informação segue da camada de saída até a camada de entrada.

A topologia da rede MLP durante a fase de retropropagação dos erros reduz-se a uma estrutura linear nos parâmetros, comumente chamada de rede MLP dual, ou simplesmente rede dual (PRINCIPE et al., 2000). As razões subjacentes à proposição do princípio MAXCORE nascem dessa topologia.

Definição 4.1 - Seja uma rede MLP com $p+1$ unidades de entrada, Q neurônios ocultos e M neurônios de saída. Assim, a dinâmica da rede dual para esta arquitetura é definida pelas seguintes expressões:

$$e_i^{(h)}(t) = \sum_{k=1}^{M} m_{ki}(t)\delta_k^{(o)}(t) \qquad \text{e} \qquad e_j^{(i)}(t) = \sum_{i=1}^{Q} w_{ij}(t)\delta_i^{(h)}(t), \tag{4.1}$$

para $i = 0, \ldots, Q$ e $j = 0, \ldots, P$. As variáveis $e_i^{(h)}(t)$ e $e_j^{(i)}(t)$ podem ser interpretadas, respectivamente, como o erro projetado na camada oculta e o erro projetado na camada de entrada. Cada um dos outros termos das expressões mostradas acima estão definidos no Capítulo 2.

Como pode ser visto na Figura 4.1, a dinâmica da rede dual é *linear nos parâmetros*. Por exemplo, o erro projetado na camada oculta é uma combinação linear dos gradientes locais dos neurônios de saída. Adicionalmente, se a função de ativação dos neurônios de saída for linear, o erro projetado na camada oculta pode ser expresso simplesmente como a combinação linear dos erros na camada de saída, ou seja

$$e_i^{(h)}(t) = \sum_{k=1}^{M} m_{ki}(t)e_k^{(o)}(t). \tag{4.2}$$

Outro ponto importante a ressaltar envolve a observação de que os procedimentos tradicionais de seleção de modelos neurais não se aproveitam do fato de a dinâmica da

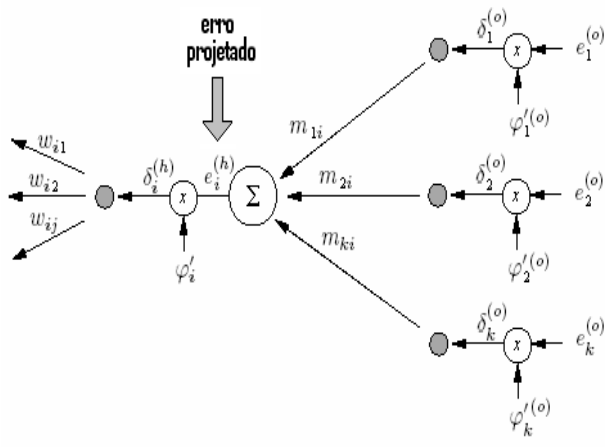

Figura 4.1: Rede Dual: topologia linear nos parâmetros.

rede dual ser linear nos parâmetros. Esta propriedade permite que uma gama de métodos e técnicas clássicas oriundas da Álgebra Linear e da Estatística possam ser explorados com o objetivo de encontrar topologias ótimas para modelos neurais.

Tendo em mente esse fato, esta tese propõe-se a explorar este novo leque de possibilidades que ora se apresenta. Um primeiro passo nesta direção é dado através da proposição do princípio Maxcore, que é enunciado a seguir.

Enunciado: Princípio da Máxima Correlação dos Erros (MAXCORE)

> Conexões sinápticas relevantes tendem a produzir maiores correlações entre os erros associados com os neurônios de uma determinada camada da rede MLP e os erros projetados na camada anterior. Conexões sinápticas não-relevantes, por outro lado, estão associadas a correlações de menor magnitude.

A idéia subjacente ao princípio Maxcore reside na percepção de que os erros associados a neurônios de uma camada estão fortemente correlacionados com os erros dos neurônios da camada anterior. Isto é uma conseqüência direta da Equação (4.1), que expressa a dependência linear entre os erros. Uma primeira tentativa de se obter métodos quantitativos baseados no princípio Maxcore é desenvolvida a seguir.

4.2 O Método CAPE

O procedimento descrito nesta seção é um *método de poda*, isto é, o usuário deve inicialmente treinar a rede MLP com um número relativamente grande de neurônios na camada escondida, e então descartar os pesos *desnecessários* ou *redundantes* com a aplicação do procedimento. É assumido que a rede foi treinada para apresentar o menor valor possível de ε_{train}. Para uma rede sobredimensionada em parâmetros, esse procedimento certamente levará a um modelo sobreajustado, com desempenho pobre na generalização. Todavia, o raciocínio é que o procedimento de poda eliminará conexões redundantes, melhorando assim o desempenho da rede na generalização.

4.2.1 Poda dos Pesos Entre as Camadas Escondida e de Saída

O procedimento de poda começa com a submissão do conjunto de treinamento uma vez mais à rede treinada. Nenhuma adaptação de pesos ocorre neste estágio. Uma vez que todos os N exemplos de treinamento tenham sido apresentados, construa as seguintes matrizes de erro

$$\mathbf{E}_o = \begin{bmatrix} e_1^{(o)}(1) & e_2^{(o)}(1) & \cdots & e_M^{(o)}(1) \\ e_1^{(o)}(2) & e_2^{(o)}(2) & \cdots & e_M^{(o)}(2) \\ \vdots & \vdots & \vdots & \vdots \\ e_1^{(o)}(N) & e_2^{(o)}(N) & \cdots & e_M^{(o)}(N) \end{bmatrix}_{N \times M} \tag{4.3}$$

e

$$\mathbf{E}_h = \begin{bmatrix} e_0^{(h)}(1) & e_1^{(h)}(1) & \cdots & e_Q^{(h)}(1) \\ e_0^{(h)}(2) & e_1^{(h)}(2) & \cdots & e_Q^{(h)}(2) \\ \vdots & \vdots & \vdots & \vdots \\ e_0^{(h)}(N) & e_1^{(h)}(N) & \cdots & e_Q^{(h)}(N) \end{bmatrix}_{N \times (Q+1)} . \tag{4.4}$$

As linhas da matriz $\mathbf{E}_o$ correspondem aos erros gerados pelos neurônios de saída para um dado exemplo de treinamento. Então, esta matriz é denominada a partir de agora como *a matriz dos erros de saída*, em contraste com a matriz $\mathbf{E}_h$, cujas linhas correspondem aos erros retropropagados associados aos neurônios da camada escondida. Em particular, a primeira coluna de $\mathbf{E}_h$ corresponde aos erros retropropagados associados aos limiares $m_{k0} = \theta_k^{(o)}$, $k = 1, \ldots, M$.

O segundo passo consiste em calcular o seguinte produto de matrizes

$$\mathbf{C}_{oh} = \mathbf{E}_o^T \mathbf{E}_h, \tag{4.5}$$

no qual T denota o transposto de uma matriz. Note que o (k,i)-ésimo elemento de $\mathbf{C}_{oh}$, denotado por $\mathbf{C}_{oh}[k,i]$, corresponde ao produto escalar (correlação) da k-ésima coluna de $\mathbf{E}_o$ com a i-ésima coluna de $\mathbf{E}_h$,

$$\mathbf{C}_{oh}[k,i] = \sum_{t=1}^{N} e_k^{(o)}(t) e_i^{(h)}(t), \tag{4.6}$$

para $k = 1, \ldots, M$, e $i = 0, \ldots, Q$.

O terceiro passo requer a organização dos valores absolutos dos elementos $\mathbf{C}_{oh}[k,i]$ em ordem crescente

$$|\mathbf{C}_{oh}[\mathbf{r}_1]| < |\mathbf{C}_{oh}[\mathbf{r}_2]| < \cdots < |\mathbf{C}_{oh}[\mathbf{r}_L]|, \tag{4.7}$$

cujo vetor $\mathbf{r}_l = (k_l, i_l)$ contém as coordenadas da posição ocupada pelo elemento l na fila, e $L = \dim(\mathbf{C}_{oh}) = M \times (Q+1)$ é o número de elementos em $\mathbf{C}_{oh}$.

O quarto passo envolve a execução do procedimento de poda mostrado na Tabela 4.1. Nesta tabela, J_{train} representa um dado índice usado para avaliação do desempenho da rede no conjunto de dados de treinamento, tal como o erro médio quadrado ε_{train} ou a taxa de classificação CR_{train}. A constante J_{tol} é um valor definido pelo usuário.

Tabela 4.1: Procedimento de poda para pesos entre as camadas escondida e de saída.

1.	Faça $l = 1$;	`// atribua 1 ao índice de contagem`
2.	**ENQUANTO** $l \leq L$ **FAÇA**	`// comece o ciclo de poda`
	2.1. Faça $a = m_{\mathbf{r}_l}$;	`// salve o valor corrente`
	2.2. Faça $m_{\mathbf{r}_l} = 0$;	`// atribua zero ao peso`
	2.3. Calcule J_{train};	`// índice de desempenho no conjunto de treinamento`
	2.4. **SE** $J_{train} < J_{tol}$,	
	ENTÃO	
	Faça $m_{\mathbf{r}_l} = a$;	`// recupere o valor`
	FIM DO SE	
	2.5. Faça $l = l + 1$;	`// continue a poda`
	FIM DO ENQUANTO	

Se $J_{train} = CR_{train}$, então J_{tol} é a taxa de reconhecimento mínima aceita para o conjunto de treinamento. Assim, elimina-se uma dada conexão $m_{\mathbf{r}_l}$ apenas se o valor de J_{train}, calculado após a eliminação da mesma, permanece acima de um valor previamente especificado de J_{tol}.

No caso em que $J_{train} = \varepsilon_{train}$, então J_{tol} é um valor de erro máximo permitido para o conjunto de dados de treinamento e a expressão condicionante na linha 2.4 da Tabela 4.1 passa a ser $J_{train} > J_{tol}$. Nesse caso, elimina-se uma dada conexão $m_{\mathbf{r}_l}$ apenas se o valor de J_{train}, calculado após a eliminação da mesma, permanece abaixo de J_{tol}.

4.2.2 Poda de Pesos Entre as Camadas de Entrada e Escondida

Para a poda dos pesos que conectam as unidades de entrada aos neurônios da camada escondida, w_{ij}, é necessário retropropagar os erros relacionados aos neurônios da camada escondida, $e^{(h)}(t)$, com o objetivo de obter os erros projetados nas unidades de entrada

$$e_j^{(i)}(t) = \sum_{i=1}^{Q} w_{ij}(t)\delta_i^{(h)}(t), \quad j = 0, \ldots, P. \tag{4.8}$$

Após a apresentação dos N vetores de treinamento, os erros resultantes podem ser organizados numa matriz de erros

$$\mathbf{E}_i = \begin{bmatrix} e_0^{(i)}(1) & e_1^{(i)}(1) & \cdots & e_P^{(i)}(1) \\ e_0^{(i)}(2) & e_1^{(i)}(2) & \cdots & e_P^{(i)}(2) \\ \vdots & \vdots & \vdots & \vdots \\ e_0^{(i)}(N) & e_1^{(i)}(N) & \cdots & e_P^{(i)}(N) \end{bmatrix}_{N\times(P+1)} . \tag{4.9}$$

Similarmente ao procedimento descrito na seção anterior, deve-se calcular o seguinte produto de matrizes

$$\mathbf{C}_{hi} = (\mathbf{E}_h^-)^T \mathbf{E}_i, \tag{4.10}$$

onde a matriz $\mathbf{E}_h^-$ é o resultado da remoção da primeira coluna da matriz $\mathbf{E}_h$. Isto se deve ao fato de que não há conexões entre os limiares dos neurônios pertencentes à camada escondida e qualquer elemento pertencente à camada de entrada. O (i, j)-ésimo elemento de $\mathbf{C}_{hi}$, denotado por $\mathbf{C}_{hi}[i, j]$, é dado por

$$\mathbf{C}_{hi}[i, j] = \sum_{t=1}^{N} e_i^{(h)}(t) e_j^{(i)}(t), \tag{4.11}$$

para $i = 1, \ldots, Q$, and $j = 0, \ldots, P$. Então, os valores absolutos dos elementos $\mathbf{C}_{hi}[i, j]$ devem ser organizados em ordem crescente

$$|\mathbf{C}_{hi}[\mathbf{s}_1]| < |\mathbf{C}_{hi}[\mathbf{s}_2]| < \cdots < |\mathbf{C}_{hi}[\mathbf{s}_L]|, \tag{4.12}$$

onde o vetor $\mathbf{s}_l = (i_l, j_l)$ contém as coordenadas do elemento ocupante da posição l na fila, e $L = \dim(\mathbf{C}_{hi}) = Q \times (P + 1)$ é o número de elementos em $\mathbf{C}_{hi}$.

O passo final envolve a execução do procedimento de poda mostrado na Tabela 4.2.

Um aspecto importante da aplicação do método proposto é que a estrutura neural deve ser submetida ao algoritmo de poda sucessivas vezes, até que não haja mais pesos a

Tabela 4.2: Procedimento de poda para pesos entre as camadas de entrada e escondida.

1.	Faça $l = 1$;	`// atribua 1 ao índice de contagem`
2.	**ENQUANTO** $l \leq L$ **FAÇA**	`// comece o ciclo de poda`
	2.1. Faça $b = w_{\mathbf{r}_l}$;	`// salve o valor corrente`
	2.2. Faça $w_{\mathbf{r}_l} = 0$;	`// atribua zero ao peso`
	2.3. Calcule J_{train};	`// índice de desempenho no conjunto de treinamento`
	2.4. **SE** $J_{train} < J_{tol}$,	
	ENTÃO	
	Faça $w_{\mathbf{r}_l} = b$;	`// recupere o valor`
	FIM DO SE	
	2.5. Faça $l = l + 1$;	`// continue a poda`
	FIM DO ENQUANTO	

serem podados (fluxograma na Figura 4.2). Isto se justifica pelo fato de que a poda de uma ou mais conexões sinápticas de um determinado neurônio oculto, por exemplo, altera o posicionamento do seu respectivo hiperplano, o que pode tornar redundante a presença de outras conexões sinápticas ou até mesmo de outros neurônios ocultos.

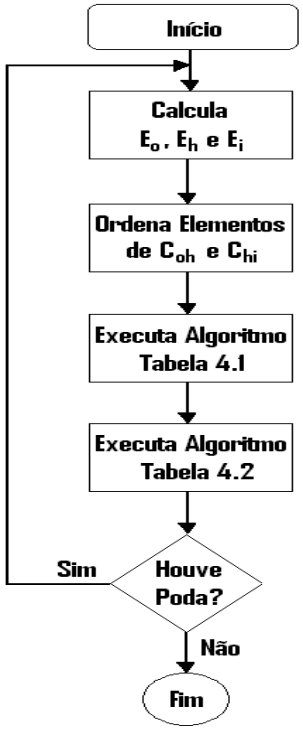

Figura 4.2: Fluxograma de aplicação do algoritmo CAPE.

O procedimento descrito nesta seção foi limitado à aplicação em uma rede MLP com apenas uma camada escondida. Entretanto, o CAPE pode facilmente ser estendido para aplicação em redes com duas ou mais camadas escondidas.

4.3 Simulações

Os experimentos apresentados nesta seção seguem um padrão no que se refere aos ajustes preliminares ao treinamento de MLPs:

- A seleção dos vetores de entrada para compor os conjuntos de treinamento, teste e, eventualmente, validação, foi realizada aleatoriamente, mas com o cuidado de contemplar cada um destes conjuntos com amostras de todas as classes.
- Os pesos e limiares foram inicializados aleatoriamente dentro de uma faixa entre $-0,5$ e $+0,5$.
- Todos os neurônios usaram função de ativação tangente hiperbólica.
- Os rótulos de saída tradicionalmente usados como $+1$ e -1 são substituídos por $+0,98$ e $-0,98$.
- Toda experimentação computacional foi realizada com o software MATLAB 7.0.

4.3.1 Simulações Preliminares

Conforme apresentado na seção 4.2, o procedimento de poda proposto deve ser implementado em duas etapas para cada aplicação do método à rede. Na concepção do método, levou-se em consideração que apenas o erro de saída é realmente observável, enquanto os ditos erros retropropagados são apenas estimativas de erro que, em sua formação, dependem de aproximações lineares das sigmóides dos neurônios em torno do ponto de operação. Dessa forma, quanto mais próximos forem os neurônios da camada de saída, mais fiéis serão suas estimativas de erro.

A seguir é apresentado um estudo comparativo da aplicação do CAPE a um conjunto de dados gerado artificialmente para ilustrar a eficiência da poda em duas etapas. O conjunto de dados possui 200 padrões (duas características) representativos de duas classes não linearmente separáveis. Cada classe possui 100 padrões, 70% dos quais utilizados para treinamento e o restante para teste. Uma rede com duas unidades de entrada, 7 neurônios ocultos e 1 neurônio de saída foi treinada com funções sigmoidais hiperbólicas até atingir uma taxa de acerto no conjunto de treinamento de $CR_{train} = 100\%$. O erro atingido no conjunto de treinamento é de $\varepsilon_{train} = 0,0005$. Já no conjunto de teste, $CR_{test} = 98,33$ e $\varepsilon_{test} = 0,0396$. Os rótulos de saída são $-0,98$ para a classe 1 e $0,98$ para a classe 2.

As matrizes de pesos alcançadas ao final do treinamento são as seguintes:

$$\mathbf{W}_1 = \begin{bmatrix} 0,5021 & 0,1669 & -0,5964 \\ -0,5539 & 0,0845 & -0,7150 \\ -0,4122 & -1,1034 & -1,1285 \\ 0,4224 & 0,2441 & -0,5318 \\ 0,0470 & 1,2423 & 1,1702 \\ 0,1899 & 0,8403 & -0,8750 \\ 0,1649 & 0,0517 & 1,1023 \end{bmatrix} \tag{4.13}$$

e

$$\mathbf{W}_2 = \begin{bmatrix} 1,5309 & 1,9338 & 2,5582 & 1,2222 & -2,1314 & 3,2617 & 1,6144 & 3,9170 \end{bmatrix} \tag{4.14}$$

Após o cálculo das matrizes de correlação $\mathbf{C}_{oh}$ e $\mathbf{C}_{hi}$, ordenou-se seus elementos em ordem crescente a partir de duas estratégias diferentes. Na primeira o ordenamento levou em consideração todos os elementos das duas matrizes . Na segunda, foi implementado o algoritmo apresentado na seção 4.2. A poda foi implementada e os resultados para ambas as estratégias são apresentados e avaliados a seguir.

A poda com ordenamento simples gerou as matrizes de pesos (4.15) e (4.16). Nestas, o número de neurônios ocultos caiu de 7 para 6 neurônios, e o número de pesos eliminados é 10. A Tabela 4.3 mostra que foram necessárias duas rodadas do algoritmo de poda para a eliminação de todos os pesos considerados redundantes para uma perfeita (100%) classificação dos dados do conjunto de treinamento. Ficou evidente que o desempenho da rede podada piorou significantemente no que se refere ao conjunto de teste.

$$\mathbf{W}_1(podada) = \begin{bmatrix} 0,5021 & 0 & -0,5964 \\ 0 & 0 & -0,7150 \\ -0,4122 & -1,1034 & -1,1285 \\ 0 & 0 & 0 \\ 0 & 1,2423 & 1,1702 \\ 0,1899 & 0,8403 & -0,8750 \\ 0 & 0 & 1,1023 \end{bmatrix} \tag{4.15}$$

$$\mathbf{W}_2(podada) = \begin{bmatrix} 1,5309 & 1,9338 & 2,5582 & 0 & -2,1314 & 3,2617 & 1,6144 & 3,9170 \end{bmatrix} \tag{4.16}$$

Tabela 4.3: Resultados de poda com ordenamento simples de pesos.

rodada	seq.	matriz	linha	coluna	CR_{train}	ε_{train}	CR_{test}	ε_{test}
1	1	1	4	1	100	0,0024	98,33	0,0330
	2	1	7	1	100	0,0033	98,33	0,0319
	3	1	2	1	100	0,0039	96,67	0,0548
	4	1	1	2	100	0,0046	96,67	0,0601
	5	1	7	2	100	0,0048	96,67	0,0617
	6	1	2	2	100	0,0067	95,00	0,0691
	7	2	1	4	100	0,0097	95,00	0,0704
	8	1	4	3	100	0,0097	95,00	0,0704
2	9	1	5	1	100	0,0097	95,00	0,0701
	10	1	4	2	100	0,0097	95,00	0,0701

A poda baseada no procedimento padrão do CAPE gerou as matrizes de pesos (4.17) e (4.18). Nota-se que a estrutura final também necessita apenas de 6 neurônios ocultos. A poda dos 9 pesos foi realizada com apenas uma rodada do algoritmo. Os resultados apresentados na Tabela 4.4 mostram que a poda foi realizada com bastante sucesso tanto no que se refere ao conjunto de dados de treinamento quanto ao conjunto de teste, pois houve a preservação das taxas de acerto.

$$\mathbf{W}_1(podada) = \begin{bmatrix} 0,5021 & 0,1669 & -0,5964 \\ 0 & 0,0845 & -0,7150 \\ -0,4122 & -1,1034 & -1,1285 \\ 0 & 0 & 0 \\ 0 & 1,2423 & 1,1702 \\ 0 & 0,8403 & -0,8750 \\ 0 & 0 & 1,1023 \end{bmatrix} \tag{4.17}$$

$$\mathbf{W}_2(podada) = \begin{bmatrix} 1,5309 & 1,9338 & 2,5582 & 0 & -2,1314 & 3,2617 & 1,6144 & 3,9170 \end{bmatrix} \tag{4.18}$$

Embora as duas estratégias eliminem a maioria dos pesos em comum, ambas percorrem caminhos diferentes no espaço de pesos e apresentam resultados finais significativamente diferentes. A Figura 4.3 mostra a evolução do erro no conjunto de treinamento em função da poda de cada peso. Pode-se ver em azul a evolução do erro quando se utiliza o ordenamento simples dos coeficientes de correlação relativos aos pesos (Tabela 4.3). Nota-se que 6 pesos da camada oculta são eliminados antes que o peso da camada de saída (circundado) seja eliminado. Já no caso da linha vermelha, a qual representa a evolução do erro quando se utiliza o duplo ordenamento (Tabela 4.4), o primeiro peso a ser eliminado é

Tabela 4.4: Resultados de poda com duplo ordenamento de pesos.

rodada	seq.	matriz	linha	coluna	CR_{train}	ε_{train}	CR_{test}	ε_{test}
1	1	2	1	4	100	0,0040	98,33	0,0349
	2	1	4	1	100	0,0040	98,33	0,0349
	3	1	7	1	100	0,0046	98,33	0,0337
	4	1	2	1	100	0,0036	96,67	0,0531
	5	1	5	1	100	0,0038	96,67	0,0529
	6	1	6	1	100	0,0056	98,33	0,0387
	7	1	4	2	100	0,0056	98,33	0,0387
	8	1	7	2	100	0,0059	98,33	0,0397
	9	1	4	3	100	0,0059	98,33	0,0397

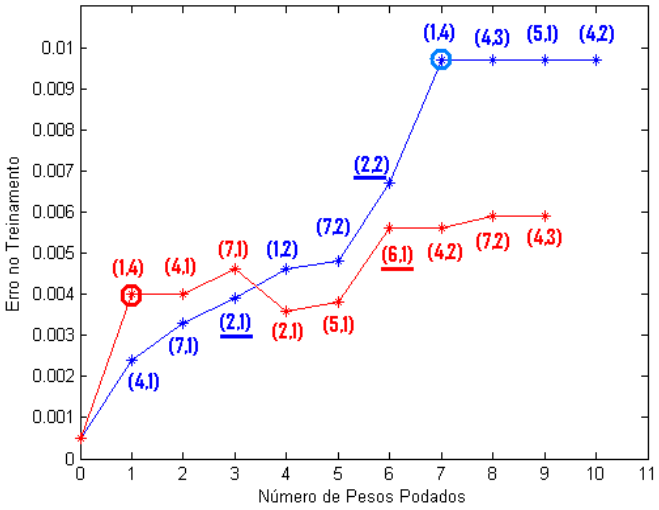

Figura 4.3: Evolução do erro no conjunto de treinamento em função dos pesos podados com ordenamento simples (azul) e ordenamento duplo (vermelho).

o peso pertencente à camada de saída (elemento que ocupa a linha 1 e coluna 4 da matriz W_2). A eliminação deste peso elimina automaticamente todos os pesos representados na linha 4 da matriz W_1. O melhor desempenho em relação ao acerto no conjunto de teste apresentado pelo procedimento padrão não se deve ao simples fato de ter podado menos pesos, mas sim pelo melhor caminho percorrido no espaço de pesos durante a poda. Isto indica que a sequência de poda é muito importante para o desempenho final da rede podada.

As superfícies de decisão geradas pelas redes podadas a partir das duas estratégias adotadas podem ser vistas nas Figuras 4.4 e 4.5.

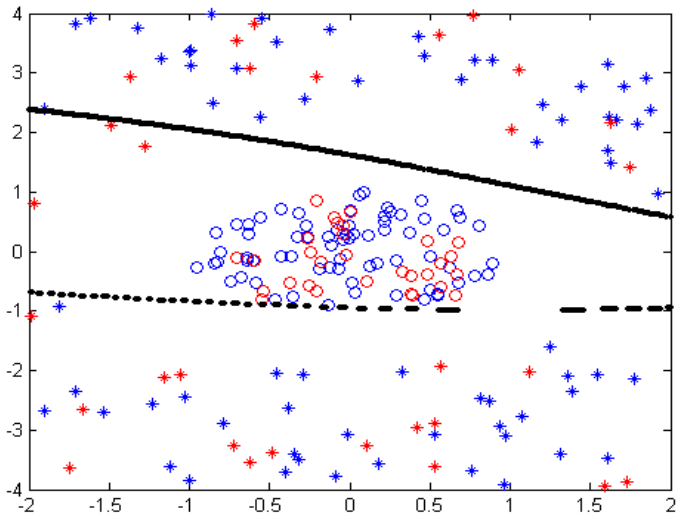

Figura 4.4: Superfície de decisão de MLP podada a partir do ordenamento simples dos coeficientes de correlação. Dados de treinamento em azul e dados de teste em vermelho.

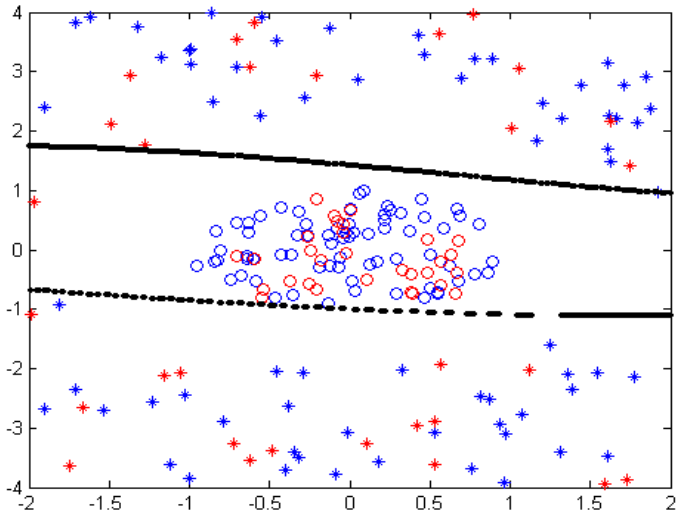

Figura 4.5: Superfície de decisão de MLP podada a partir do duplo ordenamento dos coeficientes de correlação. Dados de treinamento em azul e dados de teste em vermelho.

4.3.2 Poda em Classificadores de Padrões

Um outro conjunto de dados, com vetores de amostra bidimensionais e representando duas classes não linearmente separáveis, foi gerado artificialmente com o propósito de permitir a visualização dos hiperplanos de cada neurônio escondido e da superfície de decisão global resultante. Um total de 140 padrões foram escolhidos aleatoriamente, dentre

os 200 padrões gerados, para compor o conjunto de treinamento. O restante foi usado no conjunto de teste. A topologia inicial usada foi uma rede MLP totalmente conectada com apenas uma camada escondida ($P = 2$, $Q = 12$, $M = 1$). Pelas características do problema de classificação, o número de neurônios ocultos foi escolhido de tal forma que se tenha clara redundância. Os rótulos de saída são $-0,98$ para a classe 1 e $0,98$ para a classe 2. A Rede foi treinada com uma taxa de aprendizagem de $\eta = 0,001$ até que o acerto no treinamento se estabilizasse em $CR_{train} = 100\%$.

A aplicação sucessiva do método CAPE foi realizada conforme apresentado na seção 4.2, e os resultados numéricos são mostrados na Tabela 4.5. Nesta tabela, N_c é o número de conexões e AIC representa o critério de informação de Akaike dado pela Equação (3.18).

Tabela 4.5: Resultados da aplicação sucessiva de poda.

	Q	N_c	CR_{train}	CR_{test}	ε_{train}	ε_{test}	AIC
Arquitetura 1	12	49	100	100	0,0090	0,0098	107,42
CAPE	9	29	100	100	0,0239	0,0231	65,47
OBS	9	29	100	100	0,0239	0,0231	65,47
Arquitetura 2	8	33	100	100	0,0088	0,0153	75,47
CAPE	7	22	100	100	0,0139	0,0119	52,55
OBS	7	22	100	100	0,0139	0,0119	52,55
Arquitetura 3	6	25	100	100	0,0075	0,0110	59,79
CAPE	4	17	100	100	0,0120	0,0087	42,85
OBS	4	17	100	100	0,0120	0,0087	42,85
Arquitetura 4	4	17	100	100	0,1385	0,1440	37,95
CAPE	3	13	100	100	0,1383	0,1458	29,96
OBS	3	13	100	100	0,1383	0,1458	29,96

Para efeito de comparação de desempenho na poda, aplicou-se também o método OBS. Inicialmente, os dois métodos foram aplicados sobre a chamada Arquitetura 1 ($Q = 12$) com $N_c = 49$ pesos tendo como critério de poda a taxa de acerto no conjunto de dados de treinamento ($CR_{train} = 100\%$). Ambos os algoritmos podaram a rede de tal forma que a estrutura resultante apresenta apenas 9 neurônios ocultos, porém com menos conexões sinápticas do que teria uma rede MLP totalmente conectada com $Q = 9$ e $N_c = 37$. Embora o critério de poda imposto tenha sido a manutenção de uma taxa de acerto no conjunto de treinamento de 100%, sem qualquer imposição relativa ao conjunto de teste, o desempenho neste último foi preservado. O valor do AIC é menor para a estrutura resultante, o que indica que a mesma apresenta melhor relação entre aproximação do mapeamento entrada-saída e complexidade do modelo do que a topologia original.

Com a rede podada, foram necessários apenas 9 hiperplanos no espaço expandido

pelos neurônios ocultos para que se preservasse o desempenho da rede original, e ainda mais, o número de pesos restantes ($N_c = 29$) é menor que o de uma rede MLP totalmente conectada com $Q = 8$ neurônios ocultos e ($N_c = 33$) pesos. Isto sugere que este problema de classificação ainda pode estar sendo tratado com complexidade em demasia. Assim, outra rede MLP(Arquitetura 2) totalmente conectada com apenas uma camada escondida e com $Q = 8$ neurônios ocultos (8 hiperplanos) foi treinada novamente sob os mesmos critérios impostos ao treinamento que resultou na Arquitetura 1. Em seguida aplicaram-se os métodos de poda e, como pode ser visto na Tabela 4.5, mais uma vez os dois métodos resultaram em estruturas idênticas ($Q = 7$, $N_c = 22$) e preservaram as taxas de classificação.

Novamente, ainda parece haver excesso de complexidade na rede, e então, o processo descrito no parágrafo anterior é repetido até que se alcance a menor topologia que atenda à especificação. A Figura 4.6 mostra a evolução da topologia da rede MLP durante a busca por uma estrutura mínima com a aplicação sucessiva do método CAPE.

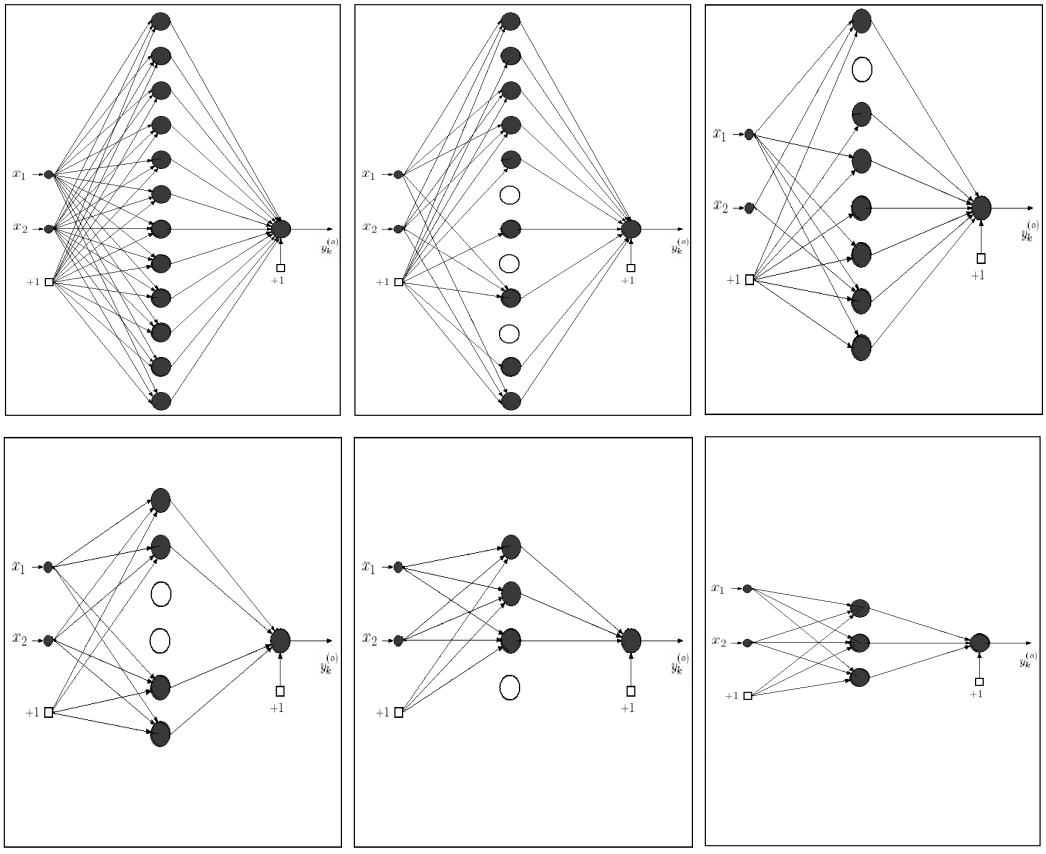

Figura 4.6: O efeito evolutivo do processo de poda sobre as arquiteturas.

Os resultados apresentados na Tabela 4.5 mostram que, pelo menos neste exemplo, o método proposto apresenta desempenho idêntico ao OBS, método este já consagrado. E

ainda mais, o procedimento aplicado pode ser utilizado para seleção de modelos neurais, o que é ratificado pela análise do AIC de todos os modelos utilizados. Pode-se ver na Figura 4.7 a evolução do AIC durante o processo de busca pela topologia mínima. As linhas escura e clara mostram a evolução do AIC nas arquiteturas treinadas e podadas, respectivamente. Apesar de ter havido uma degradação significativa de ε_{train} na topologia final ($Q = 3$), o baixo número de pesos ($N_c = 13$) contribui para o melhor compromisso entre representatividade e complexidade do modelo.

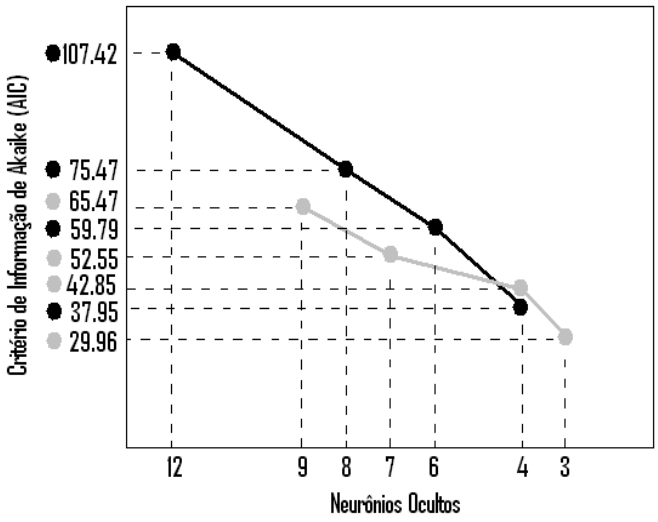

Figura 4.7: Evolução do critério do AIC nas arquiteturas treinadas (linha escura) e podadas (linha clara) durante a busca pela topologia mínima.

A Figura 4.8 mostra a superfície de decisão e o posicionamento dos hiperplanos associados à arquitetura original (Arquitetura 1). Um olhar atento à distribuição dos dados, bem como à separação das classes, leva a crer que a Arquitetura 1 apresenta complexidade demasiada para a solução do problema em questão, o que foi detectado com a aplicação progressiva dos métodos de poda e é mostrado na Figura 4.9.

É importante salientar que o OBS foi adaptado, em sua aplicação, para a comparação com o CAPE. O cálculo das sensibilidades foi preservado, mas o critério de poda foi modificado. Aqui, por questão de uniformidade utilizou-se o acerto na classificação do conjunto de treinamento, ao invés do erro no mapeamento entrada-saída, como critério para poda. Além disso, na aplicação original, a cada peso eliminado o projetista deve atualizar todo o vetor de pesos usando a Equação (3.46), ou, retreinar a rede sempre que um peso ou uma pequena parte deles (3-5%) é eliminado. Até aqui, o OBS, assim como o CAPE, foi aplicado sem atualização ou retreinamento até que não houvesse mais conexões a serem podadas. Stahlberger & Riedmiller (1996) aplicam este procedimento ao OBS.

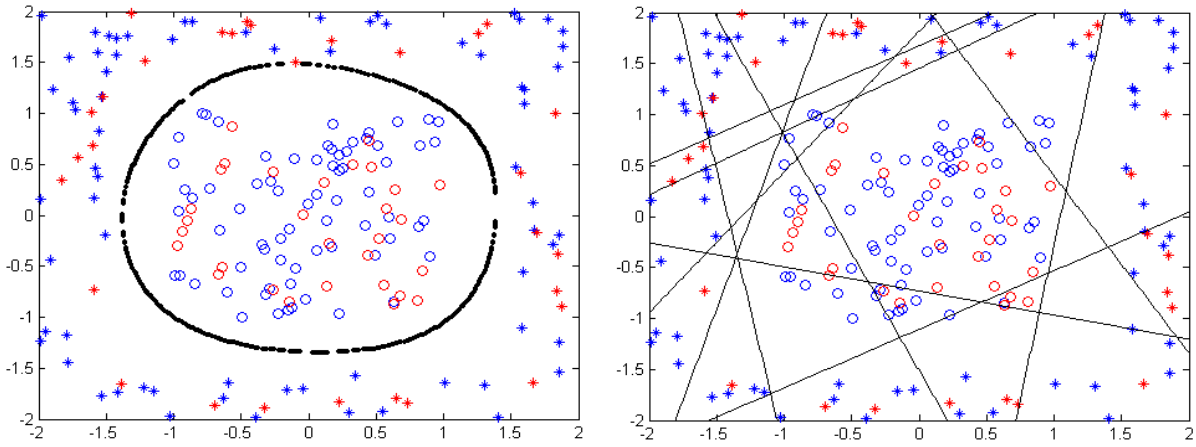

Figura 4.8: Superfície de decisão e hiperplanos para MLP treinada com $Q = 12$ neurônios ocultos. Dados de treinamento em azul e dados de teste em vermelho.

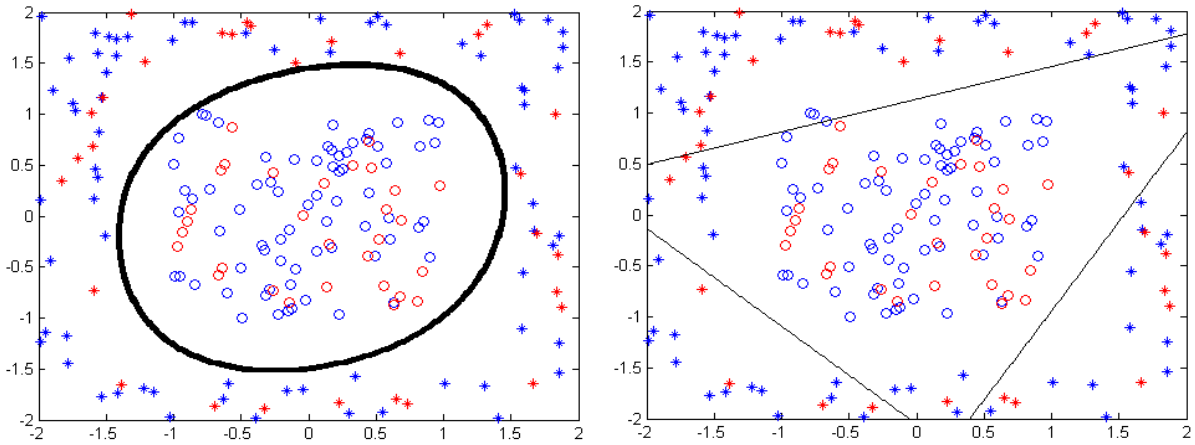

Figura 4.9: Superfície de decisão e hiperplanos para MLP com $Q = 3$ neurônios ocultos. Dados de treinamento em azul e dados de teste em vermelho.

Até aqui, foram utilizados conjuntos de dados especialmente gerados para exemplificar a aplicação do método proposto e permitir uma visualização clara de seus resultados. A partir de agora, o método será submetido a conjuntos de dados referentes a problemas reais de classificação, alguns dos quais já bem conhecidos na comunidade de redes neurais, tais como Iris, Wine e Dermatology[1].

O conjunto de dados Iris é composto por 150 padrões, cada um com 4 características (comprimento e largura da sépala, e comprimento e largura da pétala), divididos equitativamente entre 3 classes (Setosa, Versicolor e Virgínica). Os dados não foram submetidos a qualquer tipo de preprocessamento, mas com o objetivo de determinar a partição que

[1]Disponível em www.ics.uci.edu/∼mlearn. Suas características principais são apresentadas nas Tabelas E.1, E.2 e E.4 encontradas no Apêndice E.

Tabela 4.6: Resultados numéricos de poda ao conjunto Iris.

Método	Q	N_c	CR_{train}	CR_{test}	ε_{train}	ε_{test}
Modelo 1	9	75	99,05	95.56	0,0199	0,0562
CAPE(*)	9	41	99,05	95,56	0,1822	0,2153
OBS(*)	9	41	99,05	95.56	0,1849	0,2162
Modelo 2	9	75	99,05	93.33	0,0194	0,0580
CAPE(*)	7	37	99,05	95,56	0,1932	0,2309
OBS(*)	7	37	99,05	93.33	0,1884	0,2243
Modelo 3	9	75	99,05	93.33	0,0203	0,0677
CAPE(*)	6	20	99,05	93,33	0,2969	0,3144
OBS(*)	5	18	99,05	93.33	0,3444	0,3586
Modelo 4	9	75	99,05	93.33	0,0127	0,0858
CAPE(*)	4	16	99,05	93,33	0,1765	0,2422
OBS	2	11	99,05	93,33	0,2868	0,3793

propiciasse os melhores resultados de acerto no treinamento e teste, foram submetidos ao treinamento com MLPs com uma única camada oculta e com um número crescente de neurônios ocultos (Q = 5, 10 e 20) e diversas partições do conjunto de dados entre conjunto de treinamento e teste. Os melhores resultados foram obtidos com a utilização de 35 padrões por classe no conjunto de treinamento e o restante no conjunto de teste.

A Tabela 4.6 mostra os resultados da aplicação do CAPE e OBS ao conjunto de dados Iris. Foram realizados quatro treinamentos exaustivos com exatamente os mesmos dados nos conjuntos de treinamento e teste, e foram gerados quatro modelos neurais com a mesma topologia. Então, a poda foi realizada tomando-se como referência o acerto no conjunto de treinamento. Pode-se ver que, de forma geral, os métodos apresentam resultados semelhantes, mas com alguma vantagem para o OBS no que se refere ao número de pesos e neurônios finais. Observando a coluna referente ao acerto no conjunto de treinamento (CR_{train}) percebe-se que o mesmo sempre foi preservado. Já no caso do acerto no conjunto de teste, a aplicação do CAPE ao Modelo 2 produziu uma clara melhora.

Um aspecto importante, que até então não foi analisado, é o efeito da poda sobre a matriz de confusão. Por exemplo, a matriz de confusão do Modelo 4 para o conjunto de teste é a seguinte:

$$\mathbf{MC(Mod4)} = \begin{bmatrix} 33,33 & 0 & 0 \\ 0 & 26,67 & 6,67 \\ 0 & 0 & 33,33 \end{bmatrix}. \tag{4.19}$$

A primeira linha da matriz de confusão indica que $33,33\%$ de todos os dados do conjunto de teste, no caso os 15 padrões pertencentes à classe 1, foram classificados corretamente. O mesmo se deu com os dados pertencentes à classe 3. No caso da classe 2, foram classificados corretamente $26,67\%$ (12 padrões) de todos os dados pertencentes à classe 2, enquanto $6,67\%$ (3 padrões) pertencentes à classe 2 foram classificados erroneamente como pertencentes à classe 3.

A matriz de confusão sobre o conjunto de teste da rede resultante da aplicação da poda pelo método CAPE foi preservada, como se pode ver em (4.20). Mas o mesmo não se pode dizer sobre os resultados obtidos em (4.21) com o método OBS.

$$\mathbf{MC(CAPE)} = \begin{bmatrix} 33,33 & 0 & 0 \\ 0 & 26,67 & 6,67 \\ 0 & 0 & 33,33 \end{bmatrix} \tag{4.20}$$

$$\mathbf{MC(OBS)} = \begin{bmatrix} 33,33 & 0 & 0 \\ 2,22 & 26,67 & 4,44 \\ 0 & 0 & 33,33 \end{bmatrix} \tag{4.21}$$

É importante notar que, mesmo apresentando taxas iguais de acerto nos conjuntos de dados de treinamento e teste, as redes resultantes podem apresentar classificações bem distintas. Logo, o projetista deve estar atento, pois uma matriz de confusão inadequada pode ser um critério de rejeição da topologia resultante, e não somente a simples taxa de classificação.

Outro aspecto interessante observado na construção da Tabela 4.6 foi o fato de que as redes podadas cujos resultados são apresentados com uma marcação em asterisco (*) tiveram um dos neurônios de saída podados. Isto sugere uma mudança na codificação dos rótulos. A Figura 4.10 mostra o resultado de classificação do Modelo 1 ($O = 3$). O espaço de saída é tridimensional e representado por um cubo. Percebe-se que a classe Setosa está bem concentrada em um dos vértices do cubo e bastante diferenciada das outras duas classes. Já as classes Versicolor e Virgínica apresentam algum espalhamento, o que dá margem a erros de classificação. A Figura 4.11 mostra o resultado de classificação de uma versão do Modelo 1 podada com o CAPE. Esta topologia possui $N_c = 41$ pesos, $Q = 9$ neurônios ocultos e apenas $M = 2$ neurônios de saída. Os rótulos do conjunto de dados também são adaptados. Percebe-se que a eliminação de um neurônio de saída converte o espaço de saída originalmente tridimensional em bidimensional.

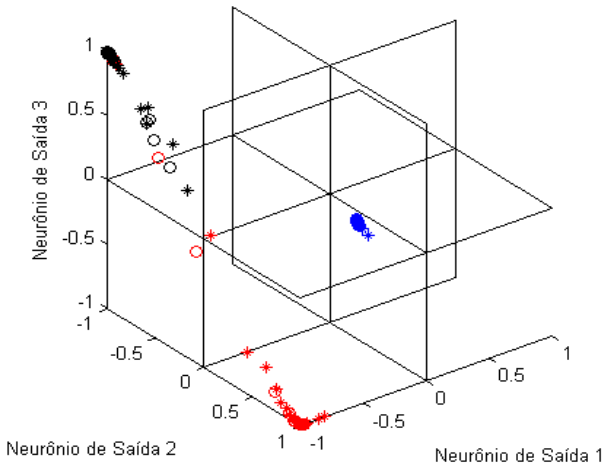

Figura 4.10: Resultados da classificação em treinamento (*) e teste (o) do conjunto de dados Iris. MLP com 3 neurônios na camada de saída.

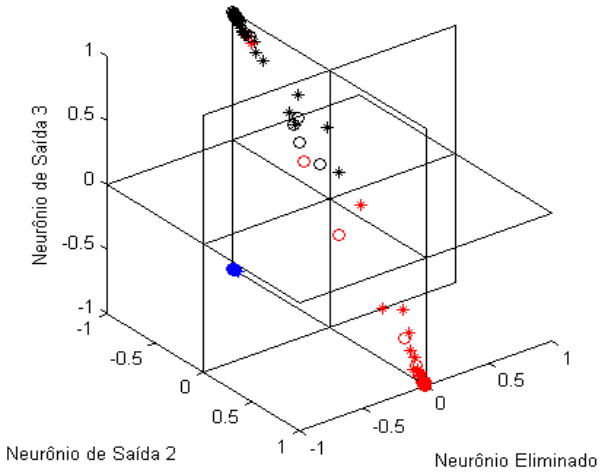

Figura 4.11: Resultados da classificação em treinamento (*) e teste (o) do conjunto de dados Iris. MLP com 2 neurônios na camada de saída.

A distância entre os rótulos das classes Versicolor e Virgínica não sofre alteração, mas entre estes e o rótulo da classe Setosa diminui para um valor 35% do original. Mesmo assim, não há erros de classificação envolvendo a classe Setosa e ainda existe uma grande margem de separação em relação às outras classes. Neste caso em particular, a mudança na codificação dos rótulos após a poda não causa maiores dificuldades de classificação. Na realidade, há uma redução do erro quadrático médio em relação à versão podada e não adaptada devido a eliminação de componentes de erro redundantes associados ao neurônio de saída podado. Os erros de treinamento e teste caem de $\varepsilon_{train} = 0,1822$ e $\varepsilon_{test} = 0,2153$ (Tabela 4.6), respectivamente, para $\varepsilon_{train} = 0,0332$ e $\varepsilon_{test} = 0,0829$.

É importante notar que a adoção desta nova rotulação para o conjunto de dados pode

criar alguma dificuldade adicional em futuros treinamentos devido a redução da distância entre os rótulos.

Foram ainda realizados 10 treinamentos com MLPs com apenas 3 neurônios na camada escondida. As taxas de acerto médio e máximo no conjunto de treinamento foram de $97,53\%$ e $98,10\%$, respectivamente. Esta redução na taxa de acerto se deve à limitada flexibilidade no posicionamento dos hiperplanos inerente de uma topologia com baixa complexidade. Já no conjunto de teste, os valores médio e máximo são, respectivamente, $94,22\%$ e $95,56\%$. A simplicidade (ou suavidade) da função de mapeamento entrada-saída aproximada pelo modelo neural favoreceu a melhores generalizações no conjunto de teste. Os procedimentos de poda foram aplicados, tomando-se como critério o acerto no conjunto de treinamento fixado em $CR_{train} = 99,05\%$. Em 60% dos casos não houve poda alguma de pesos. Entretanto, nos 40% restantes houve poda com melhora no acerto no conjunto de treinamento e com preservação do acerto no conjunto de teste. Isto é um indicativo de que a poda também pode ser benéfica para redes de pequeno porte.

4.3.3 Critérios de Poda

Nesta etapa da pesquisa pretende-se avaliar o efeito do critério de poda sobre o desempenho da rede podada em problemas de classificação. Para isso, utilizou-se um conjunto de dados biomédicos gentilmente disponibilizados pelo *Grupo de Pesquisa Aplicada em Ortopedia* (GARO) do *Centro Médico-Cirúrgico de Readaptação de Massues*, Lyon, França.

A questão consiste em classificar pacientes pertencentes a uma de três categorias: Normal (100 pacientes), Hérnia de Disco (60 pacientes) ou Spondilolistese (150 pacientes). Cada paciente é representado na base de dados por seis atributos biomédicos relacionados com a forma e orientação da pélvis e da coluna vertebral, nomeados como: incidência pélvica, inclinação pélvica, angulação sacral, raio pélvico, ângulo de lordose lombar e grau de spondilolistese.

Os seis atributos numéricos foram medidos por um especialista ortopédico a partir de imagens de raio-X da coluna vertebral. Maiores detalhes sobre esses atributos bem como sua relação com as patologias na coluna vertebral podem ser encontrados em Berthonnaud et al. (2005) e (Rocha Neto et al., 2006).

Foram realizados estudos preliminares para encontrar a partição dos dados entre conjunto de treinamento e teste que conciliasse boa representatividade dos mesmos em

Tabela 4.7: Estudo preliminar sobre o acerto percentual na aplicação de classificadores no conjunto de treinamento e teste.

Classificador	Mínimo	Médio	Máximo	Mínimo	Médio	Máximo
Perceptron Simples	33,33	64,61	82,54	58,15	73,63	84,78
Gaussiano Quadrático	26,19	77,71	92,86	14,67	77,33	91,85
Naive Bayes	71,43	81,49	89,68	73,91	82,90	89,67
MLP(Q=50)	86,51	90,71	92,06	82,07	84,13	85,33

ambos os conjuntos, além de proporcionar treinamentos consistentes e taxas de acerto compatíveis com a complexidade do problema. Assim, os dados foram pré-processados pela remoção da média e normalização pelo desvio-padrão, e em seguida, um total de 42 padrões por classe foram selecionados aleatoriamente para formarem o conjunto de treinamento. Este número refere-se 70% dos dados representativos da classe com menor número de amostras. Os 184 exemplos restantes foram usados para o propósito de testes.

Preliminarmente foram realizados algumas simulações para estabelecer referências para a determinação das taxas de acerto na classificação dos conjuntos de treinamento e teste. Os desempenhos médios de classificadores bem conhecidos como Gaussiano Quadrático, Naive Bayes e perceptron simples são apresentados na Tabela 4.7, bem como o de uma rede MLP com uma única camada escondida ($Q = 50$) treinado exaustivamente com uma taxa de aprendizagem de $\eta = 0,001$.

No caso dos classificadores bayesianos, foram realizados 1000 ensaios para compor os valores médios. Em decorrência da aplicação da regra de Bayes, o software de simulação lançou alertas de mau condicionamento da matriz de covariância dos dados quando da inversão da mesma, indicando imprecisão dos resultados. Mesmo assim, optou-se por mostrá-los. No caso do perceptron simples e da rede MLP, foram realizados 10 ensaios.

Os resultados mostram claramente que o desempenho da rede MLP na classificação do conjunto de treinamento é superior aos demais classificadores e que, embora tenha apresentado desempenho médio também superior na classificação do conjunto de teste, sua taxa de classificação máxima ficou aquém das obtidas pelos classificadores bayesianos. Entretanto, como o número de parâmetros a serem ajustados nesta MLP (503 pesos e limiares) é bem superior ao número de dados disponíveis no conjunto de treinamento (126 dados), e como a rede foi treinada até atingir a estabilização do erro médio quadrático (ε_{train}) em baixos níveis, acredita-se que esta rede MLP apresenta *overfitting*, e, conseqüentemente, taxa de acerto no conjunto de treinamento aquém do que uma rede MLP podada ou treinada com validação cruzada poderia propiciar. Então, de forma pessimista, adota-se

85% como a taxa de reconhecimento mínima aceitável para os conjuntos de treinamento e teste.

Diante do exposto, escolheu-se aleatoriamente a topologia da rede MLP totalmente conectada com $Q = 24$ neurônios ocultos e $N_c = 243$ pesos como ponto de partida para a busca de um modelo que alie baixa complexidade com boa capacidade de classificação e generalização.

Inicialmente, foram treinadas 10 destas topologias, as quais apresentaram desempenho médio de classificação de $90,08\%$ e $84,73\%$ nos conjuntos de treinamento e teste, respectivamente. E, em seguida, foram submetidas à poda, tomando como critério o acerto no conjunto de treinamento da rede original. O resultado para a aplicação do CAPE foi uma topologia que em média apresentaria o número de neurônios ocultos igual a $Q = 16,6$ e o número de pararâmetros igual a $N_c = 74,1$. No caso da aplicação do OBS, os resultados são $Q = 18,3$ e $N_c = 83,6$. Embora todas as podas tenham preservado o CR_{train} da rede original, o CR_{test} médio apresentado após a poda pelo CAPE foi de $83,37\%$ e após a poda pelo OBS foi $83,49\%$.

Na Tabela 4.8 pode ser visto, sob a denominação de "Arquitetura 1", uma das MLPs treinada e podada. Neste caso, os resultados dos acertos nos conjuntos de treinamento e teste estão confortavelmente acima do limiar de aceitação e se apresentam equilibrados entre si, ou seja, não apresentam valores muito distintos. Com a aplicação da poda com o CAPE e com o OBS, tomando como critério de poda a taxa de $89,68\%$ de acerto no conjunto de treinamento, os respectivos acertos no conjunto de teste ficaram abaixo do limiar de aceitação. Entretanto, quando se adotou o critério como 92% (marcação com "*"), obteve-se resultados satisfatórios. O CAPE podou a "Arquitetura 1" de tal forma que reduziu a camada escondida a $Q = 19$ neurônios e a $N_c = 126$ conexões. Portanto, a rede resultante apresenta o número de conexões inferior a uma rede completamente conectada com $Q = 19$ neurônios ocultos (i.e. $N_c = 193$). A aplicação do método OBS resultou numa rede podada com $Q = 22$ neurônios na camada escondida e $N_c = 156$.

Para efeito de comparação, a Tabela 4.8 também apresenta resultados da aplicação dos algoritmos de Decaimento e Eliminação de Pesos (*Weight Decay and Elimination*-WDE) e de Poda pela Magnitude do Peso (*Pruning by Weight Magnitude*-PWM). Este é outro método de poda baseado na eliminação do peso cuja magnitude é muito pequena em relação à magnitude dos demais pesos (BISHOP, 1995). O procedimento adotado é similar ao adotado pelo método CAPE. Primeiramente, ordena-se os pesos pela ordem crescente dos valores absolutos de suas magnitudes. Então, partindo do menor, realiza-se

Tabela 4.8: Resultados numéricos da aplicação dos métodos de poda.

	Q	N_c	CR_{train}	CR_{test}	ε_{train}	ε_{test}	AIC
Arquitetura 1	24	243	89,68	87,50	0,1132	0,1274	490,36
CAPE	20	111	89,68	83,70	0,2143	0,1791	225,08
OBS	21	113	89,68	78,80	0,1684	0,1940	229,56
CAPE(*)	19	126	92,06	86,96	0,1662	0,1273	255,59
OBS(*)	22	156	92,06	85,87	0,1420	0,1497	315,90
PWM	21	121	89,68	84,24	0,1733	0,1544	245,51
WDE	24	219	78,57	80,98	0,4237	0,4176	439,72

a poda dos pesos enquanto CR_{train} não seja inferior a CR_{tol}. A tabela também mostra o índice AIC, Equação (3.18), de cada modelo.

O desempenho do algoritmo WDE não foi dos melhores. Alguma experimentação foi necessária para a escolha do $\lambda = 0,001$. Além disso, o WDE mostrou-se bastante sensível aos pesos iniciais. Os resultados numéricos mostrados na Tabela 4.8 correspondem aos melhores dentre 10 tentativas de treinamento. Os métodos CAPE e OBS são muito menos sensíveis à inicialização dos pesos.

A rede que apresentou o menor índice AIC é uma resultante da aplicação do CAPE à Arquitetura 1, o que indica que este modelo apresenta o melhor compromisso entre generalização e complexidade. Entretanto, sua taxa de acerto no conjunto de teste não atingiu o limiar de aceitação. Um outro modelo, também resultante da aplicação do CAPE, foi o que apresentou os melhores resultados. É importante salientar que o modelo com o mais baixo AIC não é necessariamente o que apresenta a melhor generalização.

Costumeiramente se utiliza o conjunto de teste, o qual não participa diretamente do processo de aprendizado, para avaliar o desempenho da rede treinada. Entretanto, como o acerto no conjunto de teste é muitas vezes utilizado como parâmetro de aceitação da capacidade de generalização da rede, o conjunto de teste passa a ter influência decisiva no treinamento. Um caso claro dessa influência é o treinamento com validação cruzada. Sendo assim, optou-se por avaliar a capacidade de generalização da rede podada por outros critérios. Neste caso, o conjunto de dados foi dividido em três subconjuntos. O conjunto de treinamento contém 108 padrões, os quais representam igualmente as três classes. O conjunto de teste contém 12 padrões por classe, perfazendo um total de 36 padrões, e o conjunto de validação contém o restante (166 padrões). O acerto de classificação neste último conjunto é agora utilizado como indicador da capacidade de generalização da rede.

Mais uma vez 10 redes MLP totalmente conectadas com uma única camada escondida e com $Q = 24$ neurônios ocultos foram treinadas com o novo conjunto de treinamento.

Os resultados médios para os acertos nos conjuntos de treinamento, teste e validação, são 91,85%, 81,94% e 80,30%, respectivamente. A Tabela 4.9 traz, sob a denominação de "Arquitetura 2", os resultados de uma rede MLP típica. Percebe-se que o desempenho desta rede no acerto sobre os conjuntos de teste e validação está abaixo do limite de aceitação. Então, aplicou-se a poda pelos métodos CAPE e OBS com quatro critérios diferentes.

Tabela 4.9: Critérios para poda.

		Q	N_c	CR_{train}	CR_{test}	CR_{valid}	ε_{train}	ε_{test}	ε_{valid}
	Arquit. 2	24	243	91,67	80,56	80,12	0,0998	0,1920	0,1957
Critério 1	CAPE	20	89	91,67	80,56	77,11	0,2972	0,2817	0,3643
	OBS	20	108	91,67	75,00	78,31	0,1655	0,2214	0,2406
Critério 1	CAPE	9	33	85,19	75,00	71,69	0,3012	0,3402	0,3381
	OBS	13	46	85,19	80,56	76,51	0,2736	0,2673	0,2628
Critério 2	CAPE	7	18	75,93	86,11	78,31	0,3340	0,2912	0,2997
	OBS	7	26	79,63	86,11	80,12	0,2610	0,2423	0,2323
Critério 3	CAPE	16	52	89,81	86,11	83,13	0,2602	0,2762	0,2663
	OBS	14	66	89,81	86,11	79,52	0,2407	0,2443	0,3374
Critério 4	CAPE	14	51	87,04	86,11	87,35	0,2414	0,2568	0,2490
	OBS	13	49	85,19	86,11	80,12	0,2101	0,2302	0,2308

O critério 1 é o de acerto no conjunto de treinamento. Neste caso, o critério 1 foi aplicado duas vezes. Na primeira, a taxa de acerto foi de 91,67%, a mesma obtida ao final do treinamento, e na segunda a taxa foi de 85%, o limiar. Mais uma vez os resultados não foram satisfatórios. Na realidade, foram piores.

O critério 2 é o de acerto no conjunto de teste. Desta vez a poda promoveu uma queda nas taxas de acerto nos conjuntos de validação e treinamento. E como o conjunto de treinamento possui o número de amostras bem superior ao do conjunto de teste, o número de padrões classificados corretamente nestes dois conjuntos caíu de 128 para 113. Isto chama a atenção para uma questão importante, principalmente quando se tem poucos dados disponíveis. Sob estas condições, o projetista precisa compor o conjunto de treinamento com o maior número de amostras quanto possível. Caso contrário, pode não conseguir treinar satisfatoriamente a rede. Logo, se toda a atenção for voltada para o acerto no conjunto de teste, o acerto no conjunto de treinamento pode não apresentar bons resultados, o que pode comprometer classificações de dados desconhecidos, pois o conjunto de treinamento é composto por padrões representativos do problema.

Baseado no que foi discutido no parágrafo anterior, estabeleceu-se o critério 3. Neste, a rede é podada para classificar corretamente um mínimo de 128 padrões dentre os dados

dos conjuntos de treinamento e teste (108*0,9167+36*0,8012), e, ao mesmo tempo, manter o acerto no conjunto de teste igual ou superior a 85%. Os resultados sobre os conjuntos de treinamento e teste são menos discrepantes. No caso do conjunto de validação, a poda pelo CAPE melhorou a generalização, mas não ultrapassou o limite de aceitação.

No último critério, o critério 4, a poda é realizada de tal forma que os acertos nos conjuntos de treinamento e teste não sejam inferiores ao limite de aceitação (85%). A Tabela 4.9 mostra que a poda com o CAPE apresenta resultados consistentes, com taxas de acerto bastante equilibradas.

Selecionar modelos neurais considerando explicitamente também o desempenho da rede sobre o conjunto de teste não é exatamente uma novidade, pois Corani & Guariso (2005) usam os erros nos conjuntos de treinamento e de teste na composição da função de custo a ser minimizada durante o retreinamento de MLPs previamente podados.

4.3.4 Poda e Retreinamento

É bem sabido que o método de poda OBS, originalmente proposto em Hassibi et al. (1993), tem sido usado com um posterior retreinamento. Nesta subseção é tratado o efeito do retreinamento sobre uma rede podada.

A rede resultante da aplicação da poda pelo método CAPE a partir do critério 4 (Tabela 4.9) é tomada como referência. Inicialmente, as matrizes de pesos são reajustadas, tendo em vista que a rede foi reduzida a $Q = 14$ neurônios ocultos e $N_c = 51$ pesos, valor bem abaixo do que teria uma rede MLP totalmente conectada com 14 neurônios ocultos. Em seguida, a rede é retreinada exaustivamente, tomando os valores dos pesos resultantes da poda como ponto de partida.

A Tabela 4.10 mostra os resultados de 10 retreinamentos e posterior poda pelos métodos CAPE e OBS. Os resultados são comentados a seguir.

- O acerto no conjunto de treinamento convergiu para uma faixa relativamente estreita, enquanto o acerto no teste foi estritamente igual. Isto indica que a inicialização dos pesos com os pesos resultantes da poda precedente não dá muita margem para alteração significativa no posicionamento dos hiperplanos;
- As Arquitetura Retreinada 2 e Arquitetura Retreinada 3 apresentam resultados praticamente iguais. No entanto os resultados obtidos com a poda foram divergentes. Na realidade a Arquitetura Retreinada 3 é resultante do truncamento dos pesos da

Tabela 4.10: Resultados com retreinamento e posterior poda.

		Q	N_c	CR_{train}	CR_{test}	CR_{valid}	ε_{train}	ε_{test}	ε_{valid}
1	Arquit. Retr.	14	143	92,59	83,33	81,33	0,1050	0,1834	0,1855
	CAPE	10	44	85,19	86,11	84,34	0,1849	0,2303	0,2028
	OBS	8	35	85,19	86,11	87,95	0,1905	0,1902	0,1539
2	Arquit. Retr.	14	143	91,67	83,33	78,92	0,1014	0,1907	0,2122
	CAPE	8	33	86,11	86,11	77,11	0,2113	0,2485	0,2584
	OBS	11	41	85,19	86,11	82,53	0,2372	0,2453	0,2034
3	Arquit. Retr.	14	143	91,67	83,33	78,92	0,1014	0,1910	0,2124
	CAPE	6	30	86,11	86,11	84,34	0,1881	0,1870	0,1716
	OBS	10	42	85,19	88,89	86,75	0,2019	0,1975	0,1764
4	Arquit. Retr.	14	143	91,67	83,33	78,92	0,1013	0,1918	0,2131
	CAPE	8	35	85,19	86,11	78,92	0,1672	0,1796	0,1991
	OBS	9	37	85,19	88,89	87,95	0,2100	0,2432	0,1994
5	Arquit. Retr.	14	143	91,67	83,33	78,92	0,1023	0,1820	0,2034
	CAPE	7	36	85,19	86,11	83,13	0,2400	0,2307	0,1719
	OBS	10	37	85,19	91,67	88,55	0,1998	0,1983	0,1624
6	Arquit. Retr.	14	143	91,67	83,33	80,72	0,1013	0,1852	0,1993
	CAPE	9	33	86,11	86,11	80,72	0,2183	0,2290	0,2120
	OBS	9	36	85,19	86,11	82,53	0,2565	0,2621	0,2586
7	Arquit. Retr.	14	143	91,67	83,33	81,93	0,1004	0,1818	0,1946
	CAPE	6	29	85,19	86,11	89,16	0,2123	0,2215	0,1661
	OBS	11	46	85,19	86,11	86,14	0,2557	0,2769	0,2039
8	Arquit. Retr.	14	143	91,67	83,33	80,72	0,1013	0,1852	0,1993
	CAPE	9	33	86,11	86,11	80,72	0,2183	0,2290	0,2120
	OBS	9	36	85,19	86,11	82,53	0,2565	0,2621	0,2586
9	Arquit. Retr.	14	143	90,74	83,33	81,33	0,1047	0,1781	0,1973
	CAPE	12	47	87,04	86,11	81,93	0,1672	0,2028	0,2128
	OBS	13	61	85,19	86,11	83,13	0,1796	0,2101	0,2081
10	Arquit. Retr.	14	143	90,74	83,33	81,93	0,1034	0,1812	0,1841
	CAPE	11	38	85,19	86,11	85,54	0,1884	0,2030	0,1761
	OBS	8	31	85,19	86,11	80,72	0,2322	0,2254	0,2073

Arquitetura Retreinada 2 a quatro casas decimais. Isto ilustra a sensibilidade dos métodos de poda à precisão numérica;

- Nenhuma rede MLP retreinada produziu taxas tidas como aceitáveis no conjunto de validação;

- A aplicação posterior de poda frequentemente melhora o desempenho da rede MLP no conjunto de validação, sendo que em 5 ocasiões o OBS resultou em taxas de acerto no conjunto de validação superiores ao limite imposto, contra apenas 2 casos

alcançados pelo CAPE;

- Embora a menor topologia resultante de poda com sucesso no conjunto de validação tenha sido obtida com a aplicação do CAPE (Arquitetura Retreinada 7), o OBS foi superior em 8 ocasiões;

4.4 PCA Aplicado à Rede Dual

Conforme apresentado na subseção 3.5.1 a aplicação de PCA a MLPs no sentido direto da rede como uma ferramenta de seleção de modelos neurais é uma prática bastante investigada. Entretanto, o uso desta técnica sobre a rede dual, a qual é linear nos parâmetros, se apresenta como uma novidade.

Nesta seção são apresentados o algoritmo do PCA aplicado à rede dual e um estudo comparativo do desempenho desta técnica em relação ao CAPE. Como referência, também são apresentados resultados da aplicação do PCA à rede direta.

A aplicação de PCA é usada como uma ferramenta para estabelecer o ordenamento dos neurônios ocultos em função de suas relevâncias para a solução do mapeamento entrada-saída da rede. O algoritmo elaborado para uma rede MLP com uma camada oculta apresenta os seguintes passos:

Passo 1: Apresentar o conjunto de dados de treinamento à rede previamente treinada e calcular a matriz $\mathbf{E}_h$ (equação 4.4) relativa aos erros projetados na camada escondida;

Passo 2: Calcular a matriz de covariância ($\mathbf{C}_h$) de $\mathbf{E}_h$;

Passo 3: Calcular autovalores e autovetores de $\mathbf{C}_h$;

Passo 4: Organizar em ordem decrescente os autovalores de $\mathbf{C}_h$ baseado em seus valores absolutos e construir uma matriz de transformação $\mathbf{M}$ a partir dos autovetores de $\mathbf{C}_h$. A primeira linha de $\mathbf{M}$ é composta pelo transposto do autovetor correspondente ao autovalor que possui o maior valor absoluto, a segunda linha de $\mathbf{M}$ é composta pelo transposto do autovetor correspondente ao autovalor que possui o segundo maior valor absoluto e etc.;

Passo 5: Aplicar uma transformação à matriz $\mathbf{E}_h$ dada por

$$\mathbf{E}_h^* = \mathbf{M}.\mathbf{E}_h^T, \tag{4.22}$$

onde $\mathbf{E}_h^*$ é a matriz transformada dos erros projetados na camada escondida. A primeira linha de $\mathbf{E}_h^*$ supostamente contém as informações mais relevantes no espaço transformado;

Passo 6: Encontrar similaridade entre as linhas de $\mathbf{E}_h^*$ e as colunas de $\mathbf{E}_h$. O neurônio oculto, cujo vetor de erros projetados esteja mais correlacionado com a primeira linha de $\mathbf{E}_h^*$ (informação mais relevante no espaço transformado), é supostamente o neurônio mais relevante da camada escondida. Ao contrário, o neurônio oculto, cujo vetor de erros projetados esteja mais correlacionado com a última linha de $\mathbf{E}_h^*$ (informação menos relevante no espaço transformado), é supostamente o neurônio menos relevante da camada escondida.

Passo 7: Montar um vetor de índices de posição dos neurônios na camada oculta na ordem crescente de suas relevâncias. O primeiro elemento contém o índice de posição do neurônio oculto considerado menos relevante para a solução da rede MLP.

O algoritmo que efetivamente realiza a poda dos neurônios ocultos de uma MLP com uma camada escondida usa o vetor de índices de posição $Ip = [Ip_1\ Ip_2\ \ldots\ Ip_q\ \ldots\ Ip_Q]$ como referência para o seqüenciamento da poda. O procedimento pode ser visto na Tabela 4.11, o qual deve ser aplicado sucessivamente até que não haja mais poda de neurônio.

Tabela 4.11: Procedimento de poda de neurônios ocultos baseado na aplicação de PCA á rede dual.

1.	Faça $q = 1$;	`// atribua 1 ao índice de contagem`
2.	**ENQUANTO** $q \leq Q$ **FAÇA**	`// comece o ciclo de poda`
	2.1. Faça $a_i = m_{i,Ip_q}$; $i = 1, \ldots, O$	`// salve os pesos e bias relacionados`
	$b_j = w_{Ip_q,j}$; $j = 1, \ldots, Q+1$	`// ao neurônio a ser podado`
	2.2. Faça $m_{i,Ip_q} = 0$; $i = 1, \ldots, O$	`// atribua zero aos pesos e bias`
	$w_{Ip_q,j} = 0$; $j = 1, \ldots, Q+1$	`// relacionados ao neurônio a ser podado`
	2.3. Calcule J_{train};	`// índice de desempenho no conjunto`
		`// de treinamento`
	2.4. **SE** $J_{train} < J_{tol}$,	
	ENTÃO	
	Faça $m_{i,Ip_q} = a_i$; $i = 1, \ldots, O$	`// recupere os pesos e bias`
	$w_{Ip_q,j} = b_j$; $j = 1, \ldots, Q+1$	`// relacionados ao neurônio`
	FIM DO SE	
	2.5. Faça $q = q + 1$;	`// continue a poda`
	FIM DO ENQUANTO	

Tabela 4.12: Comparação PCA e CAPE.

Dado	Método	Q	N_c	CR_{tr}	ε_{tr}	CR_{ts}	ε_{ts}	N_Q Podados
	original	9	75	99,05	0,0203	93,33	0,0677	-
Iris	CAPE	6	20	99,05	0,2969	93,33	0,3144	4,5,9
	PCAD	6	51	99,05	0,0303	93,33	0,0751	4,5,7
	PCA	6	51	99,05	0,0303	93,33	0,0751	4,5,7
	original	12	49	100	0,0090	100	0,0098	-
Problema	CAPE	5	21	100	0,0239	100	0,0231	2,5,6,7,8,10,11
seção 4.3.2	PCAD	9	37	100	0,0206	100	0,0081	6,8,10
	PCA	9	37	100	0,0206	100	0,0081	6,8,10
	original	24	243	89,68	0,1132	87,50	0,1274	-
Coluna	CAPE	18	124	92,06	0,1662	86,96	0,1273	1,4,5,9,22
	PCAD	17	173	90,48	0,2267	87,50	0,1679	1,5,9,12,13,20,21
	PCA	20	203	89,68	0,1581	86,96	0,1365	1,4,7,9

Os resultados da aplicação de PCA à rede MLP dual são apresentados na Tabela 4.12. Nela, foram utilizados conjuntos de dados previamente testados neste trabalho e, para efeito de comparação, também são apresentados resultados referentes à aplicação do CAPE e de PCA à rede direta. Esta última metodologia se utiliza basicamente dos mesmos procedimentos do PCA aplicado à rede dual, mas é aplicada à rede direta. Aqui, a matriz $\mathbf{E}_h$, referente aos erros projetados na camada oculta, é substituída pela matriz $\mathbf{Y}_h$, de mesma dimensão, mas referente à saída dos neurônios ocultos mediante a apresentação de todo o conjunto de treinamento.

A aplicação dos métodos à rede MLP treinada para a classificação do conjunto Iris mostra plena coincidência na poda dos 3 neurônios ocultos (coluna N_Q Podados) quando se refere ao PCA aplicado à rede direta (PCA) e à rede dual (PCAD). No caso da aplicação do CAPE, também obteve-se a poda de 3 neurônios ocultos, 2 dos quais coincidentes com os resultados anteriores.

Já no caso do problema apresentado na seção 4.3.2 a aplicação do CAPE resulta na poda de 7 neurônios ocultos, 3 dos quais coincidentes com os resultados dos outros dois métodos. É importante salientar que a aplicação do CAPE resultou diretamente na poda de 3 neurônios, justamente os mesmos podados pela aplicação de PCA. Mas como restam ainda 4 neurônios ocultos conectados apenas ao *bias* (Figura 4.12), pode-se eliminar tais neurônios desde que se faça a alteração necessária ao *bias* da camada de saída.

No caso da aplicação dos métodos à rede MLP treinada com o conjunto de dados referente a doenças relacionadas com a coluna vertebral (seção 4.3.3), a poda com o PCAD resultou no descarte do maior número de neurônios ocultos. A coincidência na

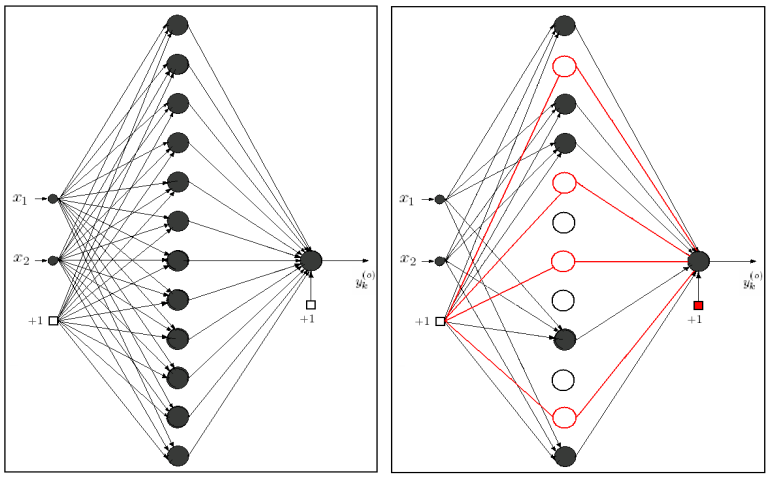

Figura 4.12: Topologias original e resultante de poda com o CAPE referente ao problema da seção 4.3.2 e apresentado na Tabela 4.12

poda de neurônios entre os três métodos ocorre apenas em 2 neurônios (1 e 9).

Nos três exemplos apresentados na Tabela 4.12 observa-se a drástica redução do número de conexões sinápticas alcançada pela aplicação do CAPE. Mesmo na aplicação dos métodos à rede MLP treinada com o conjunto de dados Coluna, quando o PCAD descartou um número maior de neurônios ocultos (7) do que o CAPE (5), o descarte de conexões sinápticas é maior com o CAPE. Entretanto, neste caso particular, a maior redução do número de neurônios ocultos obtida com o PCAD pode ter um maior impacto no custo computacional da rede podada, pois a poda de neurônios ocultos efetivamente reduz a dimensão das matrizes de pesos, reduzindo o número de operações matemáticas executadas na aplicação da rede MLP, enquanto o descarte de conexões sinápticas sem a eliminação de neurônios ocultos apenas acrescenta zeros às matrizes de pesos.

Surge uma questão importante: como avaliar a relação custo-benefício do modelo resultante da aplicação dos métodos de seleção de modelos acima citatos quando se leva em consideração a redução da redundância (ou complexidade) e a redução do custo computacional?

O custo computacional pode ser avaliado de forma simplificada como o número total de operações matemáticas executadas na aplicação da rede MLP para a apresentação de uma amostra. No caso particular de uma rede MLP com apenas uma camada escondida, o número de operações é $N_{op} = 2.(PQ + QO) + Q + O$. Aqui não há distinção entre o custo de produtos, somas ou cálculo de tangente hiperbólica.

Já no caso da redução da redundância, pode-se adaptar uma medida utilizada em Ri-

Tabela 4.13: Resultados da Comparação Entre PCA e CAPE.

Dado	Método	Q	N_c	CR_{tr}	ε_{tr}	N_{op}	AIC	$\kappa(\mathbf{J})$	$(\frac{dE}{dw})_{med}$
	original	9	75	99,05	0,0203	138	157,79	1,6042e+013	0,0898
Iris	CAPE	6	20	99,05	0,2969	93	42,43	1,0229e+013	0,3827
	PCAD	6	51	99,05	0,0303	93	108,99	1,4393e+012	0,1647
	PCA	6	51	99,05	0,0303	93	108,99	1,4393e+012	0,1647
	original	12	49	100	0,0090	85	107,42	2,5940e+010	0,0264
Problema	CAPE	5	21	100	0,0239	36	49,47	8,2354e+004	0,0911
seção 4.3.2	PCAD	9	37	100	0,0206	64	81,76	4,4866e+009	0,0573
	PCA	9	37	100	0,0206	64	81,76	4,4866e+009	0,0573
	original	24	243	89,68	0,1132	459	490,36	3,4832e+007	0,1151
Coluna	CAPE	18	124	92,06	0,1662	345	251,59	6,4178e+006	0,1990
	PCAD	17	173	90,48	0,2267	326	348,97	9,1511e+006	0,1814
	PCA	20	203	89,68	0,1581	383	409,69	4,2381e+007	0,1358

vals & Personnaz (2000) para avaliação da significância de cada conexão sináptica para a saída do modelo neural. Estes autores propuseram a busca por uma topologia de rede ótima (ou sub-ótima) partindo de uma rede MLP com poucos neurônios ocultos. A rede MLP deve ser incrementada, isto é, o número de neurônios na camada escondida deve crescer, até que se atinja uma topologia mal condicionada, ou seja, até que sua matriz Jacobiana ($\mathbf{J}$) seja mal condicionada. Como os elementos da matriz Jacobiana representam a sensibilidade da saída do modelo em relação a cada peso, o mal condicionamento desta matriz é naturalmente um sintoma de que alguns pesos são desnecessários. Mais detalhes sobre a aplicação desta metodologia para seleção de modelos neurais ainda pode ser encontrada em (RIVALS; PERSONNAZ, 2003, 2004).

O condicionamento da matriz Jacobiana é obtido a partir da sua decomposição em valores singulares e do cálculo do número de condicionamento $\kappa(\mathbf{J})$ (razão entre o maior e o menor valor singular). A matriz $\mathbf{J}$ pode ser considerada como muito mal condicionada quando $\kappa(\mathbf{J}^T\mathbf{J}) = (\kappa(\mathbf{J}))^2$ atinge o inverso da precisão numérica[2]. Considerando uma precisão numérica da ordem de 10^{-16}, $\mathbf{J}$ é considerada mal condicionada se o seu número de condicionamento superar 10^8.

A Tabela 4.13 é uma reedição da Tabela 4.12, mas agora com ênfase em medidas relacionadas ao custo computacional, representatividade do modelo e níveis de redundância. Observa-se que o compromisso representatividade do mapeamento entrada-saída *versus* complexidade do modelo, apontado pelo AIC, sugere que os modelos resultantes

[2] O treinamento de MLPs utilizado em (RIVALS; PERSONNAZ, 2000) foi baseado no algoritmo Levenberg-Marquardt, o qual depende de $(\mathbf{J}^T\mathbf{J})^{-1}$ para a adaptação dos pesos. Os resultados deste produto são confiáveis se a precisão numérica não for inferior a 10^{-16}.

da aplicação do CAPE levam clara vantagem, pois o índice é fortemente influenciado pelo número de conexões sinápticas restantes. Quanto ao custo computacional, todas as topologias resultantes da aplicação dos métodos de seleção de modelos apresentam redução do número de operações. Percebe-se também que, em geral, o número de condicionamento das topologias resultantes é menor que o da topologia original. Estas duas últimas observações sugerem que estes modelos são mais eficientes.

É importante salientar que o número de condicionamento da topologia resultante da aplicação do CAPE à rede MLP treinada com o conjunto de dados Iris é maior que os resultados obtidos com os outros dois métodos. Isto, entretanto, não implica numa menor eficiência na utilização dos pesos para alcançar os resultados de reconhecimento de padrões requeridos, pois a topologia resultante da aplicação do CAPE utiliza apenas 27% dos pesos originais, enquanto os outros dois métodos utilizam 68%. Logo, o número de condicionamento talvez não seja a melhor medida para avaliação da eliminação de redundância.

A última coluna da Tabela 4.13 refere-se ao valor médio das sensibilidades da função custo em relação aos pesos ($(\frac{dE}{dw})_{med}$). Observa-se que as topologias resultantes das podas sempre apresentam maiores valores médios de $\frac{dE}{dw}$, o que significa que os pesos mais significativos para a modelagem neural foram preservados. E ainda mais, o CAPE se sobressai neste aspecto.

4.5 Conclusões

Neste capítulo foi apresentado o algoritmo CAPE como proposta metodológica para realização de poda de pesos em MLPs.

Tratou-se inicialmente da formalização do algoritmo e, em seguida, a atenção foi voltada para a descrição e avaliação de resultados de diversas simulações computacionais envolvendo a poda de MLPs completamente conectadas. Para este propósito, foram utilizados alguns conjuntos de dados artificiais, construídos especialmente para salientar determinadas características do método proposto, e conjuntos de dados bem conhecidos, para validação da eficácia do método.

Os resultados mostram que o CAPE é uma ferramenta atrativa na seleção de modelos neurais, tendo em vista que a poda de pesos pode eventualmente evoluir para a eliminação de neurônios. Foi notado também que esta redução de estrutura traz benefícios para a capacidade de generalização da rede e diminuição do esforço computacional na

implementação de MLPs em classificadores embarcados.

O desempenho do CAPE foi principalmente comparado com o do OBS, cuja implementação é objeto do apêndice A. Desta comparação percebeu-se que, em geral, o CAPE apresenta resultados semelhantes aos obtidos com o OBS, mas com a vantagem de não fazer suposições prévias sobre o treinamento da rede e apresentar baixa complexidade na implementação. Outra vantagem importante do CAPE em relação ao OBS é o baixo custo operacional. Um estudo adicional, em que se compara os custos operacionais do CAPE em relação aos do OBS, é apresentado no apêndice D.

Foi apresentado também um algoritmo para aplicação de PCA à rede MLP dual. Seus resultados foram comparados aos alcançados pelo CAPE. Percebeu-se que os desempenhos dos dois algoritmos são comparáveis em relação à capacidade final de generalização das redes podadas, mas o CAPE apresenta larga vantagem em relação ao número de conexões sinápticas podadas.

5 *CAPE Como Ferramenta Para Extração de Características*

Ao se aplicar a poda aos pesos que conectam as unidades de entrada à primeira camada oculta, podem ocorrer situações em que todos os pesos associados a uma ou mais unidades de entrada sejam podados. Assim, pode-se pensar em usar a poda para realizar seleção de características, ou seja, para selecionar atributos relevantes do vetor de entrada. Esta estratégia é discutida em maior nível de detalhamento nas próximas seções deste capítulo.

5.1 Seleção de Características com o CAPE

A existência de redundância entre atributos em um conjunto de dados não é prejudicial. Na realidade pode até ser benéfica, pois pode reforçar tendências nos dados. O que pode ser prejudicial é o excesso de redundância, pois pode dificultar a aprendizagem, aumentar o custo computacional e comprometer o desempenho de classificadores neurais.

Para avaliar o desempenho do CAPE na seleção de características utilizou-se o conjunto de dados chamado Wine[1]. A escolha deste conjunto não foi casual, pois o mesmo é composto por 178 padrões, cada um dos quais com 13 atributos, e a existência de muitos atributos aumenta as chances de redundância entre os mesmos. O conjunto de dados contém três classes divididas em 59, 71 e 48 exemplos. Os dados foram pré-processados pela remoção da média e normalização pelo desvio-padrão, antes de serem escolhidos aleatoriamente para compor o conjunto de dados de treinamento. Ensaios preliminares indicaram que 33 amostras selecionadas aleatoriamente por classe deveriam compor o conjunto de treinamento, sendo o restante (26, 38 e 15) atribuído ao conjunto de teste. Embora o conjunto de dados não seja simétrico, houve o cuidado de realizar o treinamento sem o favorecimento numérico de qualquer classe.

[1] Wine - os dados são o resultado de análise química de vinhos produzidos numa mesma região, porém, por produtores diferentes. Os atributos referem-se a 13 características encontradas nos 3 tipos de vinho. Suas características principais são apresentadas na Tabela E.2 no Apêndice E.

Tabela 5.1: Resultados médios de treinamento e poda para o conjunto Wine.

	Q	N_c	CR_{train}	CR_{test}	N_i
Arquitetura treinada	9	156	100	95,06	13
Arquitetura podada (CAPE)	4,6	24,9	100	92,64	9,6
Arquitetura podada (OBS)	5,7	33,0	100	94,81	10,5

A Tabela 5.1 mostra valores resultantes médios da aplicação do CAPE e do OBS ao conjunto de dados Wine. A coluna N_i representa o número de atributos utilizados no treinamento ou que restaram após a aplicação da poda. Foram realizadas 10 simulações de treinamento e poda. Por conseqüência da poda com o CAPE, nenhum atributo foi eliminado em todas as simulações. O atributo 6 foi eliminado em 6 simulações, e os atributos 5 e 8 foram eliminados em 5 simulações. Na realidade, apenas o atributo 13 foi preservado em todas as simulações. A aplicação do OBS produziu números semelhantes.

Esta variabilidade na eliminação de atributos não significa inconsistência, mas sim, dependência do modelo obtido no treinamento. Por exemplo, considerando um conjunto de dados, no qual dois de seus atributos sejam correlacionados o bastante para serem considerados redundantes, é possível que um desses atributos seja eliminado a partir da poda aplicada sobre um certo modelo neural "MN1", e o outro o seja num certo modelo neural "MN2". A eliminação dos dois atributos poderia causar perda de informação e conseqüente dificuldade de classificação.

Ensaios complementares, envolvendo todo o conjunto de dados disponível como conjunto de treinamento, foram realizados e seus resultados são apresentados na Tabela 5.2. Na coluna ATR_{elim} são mostrados os atributos eliminados pela poda. Aqui, o conjunto de dados foi submetido à seleção progressiva de características. Inicialmente, treinou-se uma rede MLP com uma única camada escondida com 9 neurônios ocultos até atingir 100% de acerto. Em seguida, aplicou-se tanto o CAPE quanto o OBS para selecionar características. O CAPE encontrou 3 atributos redundantes, enquanto o OBS encontrou apenas 1.

Diante deste resultado, formou-se um novo conjunto de dados, agora com a exclusão dos atributos 6, 7 e 9, perfazendo apenas $N_i = 10$ atributos e treinou-se outra rede MLP e aplicou-se a poda. Este procedimento foi repetido progressivamente até que não houvesse mais atributos a serem eliminados. Pode-se ver claramente que o CAPE mostrou-se mais contundente na seleção de características. Ao final, restaram apenas 8 características.

Para confirmar esta habilidade como extrator de características, o método CAPE

Tabela 5.2: Resultados da seleção progressiva de características no conjunto de dados Wine.

	Q	N_c	CR_{train}	ε_{train}	N_i	ATR_{elim}
Arquitetura 1	9	156	100	0,0006	13	-
CAPE	7	37	100	0,2018	10	6,7,9
OBS	9	62	100	0,0257	12	6
Arquitetura 2	9	129	100	0,0023	10	-
CAPE	7	32	100	0,2599	9	5
OBS	8	43	100	0,0360	10	-
Arquitetura 3	9	120	100	0,0011	9	-
CAPE	5	26	100	0,0953	8	2
OBS	6	29	100	0,0717	9	-
Arquitetura 4	9	111	100	0,0011	8	-
CAPE	4	25	100	0,1100	8	-
OBS	5	28	100	0,0349	8	-

é aplicado novamente, agora sobre o conjunto de dados *Dermatology*. Os dados estão organizados em 6 grupos de 112, 61, 72, 49, 52 e 20 padrões, perfazendo um total de 366 padrões, cada um dos quais com 34 atributos, e referem-se à caracterização de 6 doenças de pele. Os dados foram pré-processados pela remoção da média e normalização pelo desvio-padrão. Todos os dados disponíveis compuseram o conjunto de dados de treinamento.

Procedimento semelhante ao aplicado ao conjunto de dados *Wine* foi aplicado ao conjunto *Dermatology*. Os resultados são apresentados na Tabela 5.3. Nesta, todas as arquiteturas foram treinadas para atingirem taxas de acerto não inferiores a $99,50\%$. Utilizou-se também $CR_{train} > 99,50\%$ como critério de poda.

Pode-se observar que, a cada rodada do procedimento, a seleção pela aplicação do CAPE é, em geral, mais agressiva, o que acelera o processo. E por isso, foi utilizada como referência para o treinamento da arquitetura subsequente.

A aplicação do método atingiu a eliminação de 15 atributos ao final da aplicação da poda à Arquitetura 7. A eficácia do método ficou evidenciada quando a rede MLP foi treinada com sucesso com os 19 atributos restantes e atingiu uma taxa de acerto de $99,73\%$, como pode ser visto na Arquitetura 8.

Fato interessante ocorreu quando da aplicação do CAPE à Arquitetura 8. A poda sugere a eliminação dos atributos 16 e 26, e mantém a taxa de acerto acima do mínimo aceitável $CR_{train} > 99,50\%$. Porém, o treinamento de uma nova arquitetura com a utilização de apenas 17 atributos não atingiu sequer o limite mínimo aceitável. Assim, treinou-se

Tabela 5.3: Resultados da seleção progressiva de características no conjunto de dados *Dermatology*.

	Q	N_c	CR_{train}	ε_{train}	N_i	ATR_{elim}
Arquitetura 1	10	416	99,73	0,0030	34	-
CAPE	7	85	99,73	0,0193	27	1,11,20,29,32,33,34
OBS	9	141	99,73	0,0298	31	1,11,30
Arquitetura 2	10	346	99,73	0,0034	27	-
CAPE	8	106	99,73	0,0607	25	17,30
OBS	10	110	99,73	0,0423	26	30
Arquitetura 3	10	326	100	0,0023	25	-
CAPE	10	121	100	0,0648	23	12,25
OBS	10	115	100	0,0595	24	16
Arquitetura 4	10	306	99,73	0,0037	23	-
CAPE	7	78	99,73	0,0580	22	18
OBS	10	110	99,73	0,0374	23	-
Arquitetura 5	10	296	99,73	0,0036	22	-
CAPE	9	104	99,73	0,0900	21	24
OBS	10	110	99,73	0,0860	22	-
Arquitetura 6	10	286	99,73	0,0032	21	-
CAPE	9	98	99,73	0,0814	20	7
OBS	9	98	99,73	0,0454	19	7,10
Arquitetura 7	10	276	99,73	0,0039	20	-
CAPE	9	97	99,73	0,0525	19	13
OBS	10	92	99,73	0,0525	19	13
Arquitetura 8	10	266	99,73	0,0042	19	-
CAPE	9	78	99,73	0,0485	17	16,26
OBS	10	111	99,73	0,0366	18	16

duas redes MLP considerando 18 atributos, com a eliminação do atributo 16 em uma e com a eliminação do atributo 26 na outra. Com a eliminação do atributo 26, a taxa de acerto mínima aceitável não foi atingida, o que descarta esta opção. Já com a eliminação do atributo 16 obteve-se sucesso. E com a poda, chegou-se a uma rede MLP com $Q = 10$, $N_c = 109$, $N_i = 18$, $CR_{train} = 99,73\%$ e $\varepsilon_{train} = 0,0382$. Diante do exposto, o conjunto de dados *Dermatology* pode ser usado, sem perda de informação, com a eliminação de 16 atributos ditos redundantes (1,7,11,12,13,16,17,18,20,24,25,29,30,32,33,34).

A redução do número de atributos alcançada nos dois problemas tratados nesta seção foi bastante significativa. Entretanto, como o algoritmo de treinamento das redes MLP utilizado até aqui é o tradicional algoritmo de retropropagação do erro, acredita-se que

a utilização de um algoritmo como o Levenberg-Marquardt[2], que também leva em consideração informação de segunda ordem da função custo (erro quadrático médio) a ser minimizada, possa dar margem a eliminação de mais atributos, principalmente no caso do conjunto de dados *Dermatology*. Dessa forma, partido das condições finais alcançadas nos dois problemas acima tratados, realizaram-se treinamentos complementares com o algoritmo Levenberg-Marquardt. Entretanto, o número de atributos remanescentes permaneceu inalterado.

5.2 Seleção de Características com a Associação CAPE-PCA: Um Novo Método

Conforme citado anteriormente, a seleção de características a partir da aplicação de métodos de poda promove a eliminação de atributos redundantes. Estes, entretanto, não são necessariamente os de menor relevância. Porém, com a ação combinada do CAPE e do PCA, pode-se alcançar uma metodologia de seleção de características mais agressiva, tendo em vista que a detecção de redundância promovida pelo CAPE está aliada à detecção de relevância promovida pelo PCA.

É importante entender os conceitos de redundância e relevância, aqui aplicados a atributos. Em problemas de classificação, como é o caso desta tese, um atributo pode ser considerado redundante quando o mesmo pode ser removido do conjunto de dados e, mesmo assim, o classificador é capaz de atingir um dado desempenho pré-especificado pelo projetista. A redundância de um atributo pode está relacionada com a ausência de informação significativa para a classificação, ou pelo fato de existir pelo menos um outro atributo que contenha informação similar e útil para a classificação. Neste último caso, o projetista pode treinar o classificador neural diversas vezes utilizando os demais atributos e um dos atributos considerados redundantes entre si, descartando o outro atributo redundante. Esta metodologia pode ser repetida, agora utilizando o atributo descartado inicialmente no conjunto de dados e excluindo o outro. O atributo que, em média, propiciar classificadores com maior capacidade de discriminação das classes é considerado o mais relevante.

[2] O algoritmo Levenberg-Marquardt atualiza iterativamente o vetor de parâmetros θ de acordo com

$$\theta_i = \theta_{i-1} + (\mathbf{J}_{i-1}^T \mathbf{J}_{i-1} + \lambda_i \mathbf{I})^{-1} \mathbf{J}_{i-1}^T (y_p - f(x, \theta_{i-1}))$$

onde $\mathbf{J}_{i-1}$ é a matriz jacobiana calculada na iteração $i-1$, λ_i é um escalar positivo e $\mathbf{I}$ é uma matriz identidade com a mesma dimensão da matriz jacobiana (RIVALS; PERSONNAZ, 2000).

A seguir, é apresentada uma aplicação de PCA, inspirado na proposição formulada por Chen et al. (2001), para estimar a relevância relativa de atributos dentro de um conjunto de dados multidimensionais. O algoritmo apresenta os seguintes passos:

Passo 1: Organizar os dados originais em uma matriz $\mathbf{X}$, onde cada coluna comporta um vetor (padrão) de k componentes;

Passo 2: Para cada linha de $\mathbf{X}$ isoladamente, calcular média e desvio padrão e normalizar cada componente pela remoção dessa média e subseqüente divisão pelo desvio padrão. A matriz normalizada deve ser denotada como $\mathbf{X}_s$;

Passo 3: Calcular a matriz de covariância ($\mathbf{C}_s$) de $\mathbf{X}_s^T$;

Passo 4: Calcular autovalores e autovetores de $\mathbf{C}_s$;

Passo 5: Organizar em ordem decrescente os autovalores de $\mathbf{C}_s$ baseado em seus valores absolutos e construir uma matriz de transformação $\mathbf{T}$ a partir dos autovetores de $\mathbf{C}_s$. A primeira linha de $\mathbf{T}$ é composta pelo transposto do autovetor correspondente ao autovalor de maior valor absoluto, a segunda linha de $\mathbf{T}$ é composta pelo transposto do autovetor correspondente ao autovalor de segundo maior valor absoluto e assim por diante;

Passo 6: Aplicar uma transformação à matriz $\mathbf{X}_s$ dada por

$$\mathbf{X}_s^* = \mathbf{T}.\mathbf{X}_s, \tag{5.1}$$

onde $\mathbf{X}_s^*$ é a matriz transformada de $\mathbf{X}_s$. A primeira linha de $\mathbf{X}_s^*$ contém as informações mais relevantes no espaço transformado;

Passo 7: Encontrar similaridade entre as linhas da matriz $\mathbf{X}_s^*$ e as linhas de $\mathbf{X}_s$. O atributo cujo vetor representativo esteja mais correlacionado com a primeira linha da matriz $\mathbf{X}_s^*$ (informação mais relevante no espaço transformado), é supostamente o atributo mais relevante. Ao contrário, o atributo cujo vetor esteja mais correlacionado com a última linha de $\mathbf{X}_s^*$ (informação menos relevante no espaço transformado), é supostamente o atributo menos relevante.

Passo 8: Montar um vetor de índices dos atributos na ordem crescente de suas relevâncias. O primeiro elemento contém o índice de posição do atributo considerado menos relevante.

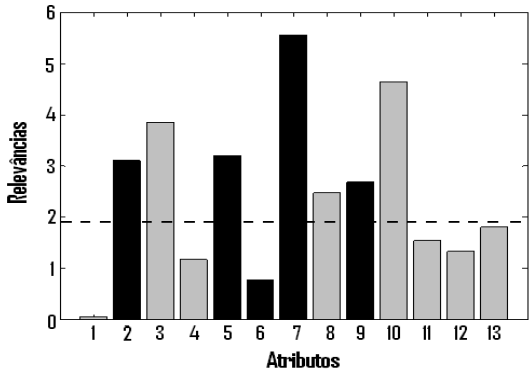

Figura 5.1: Panorama das relevâncias dos atributos do conjunto de dados *Wine* com a eliminação dos atributos (barras negras) pelo CAPE.

Este algoritmo é aplicado aos dois conjuntos de dados, *Wine* e *Dermatology*, e os seus resultados são confrontados com aqueles apresentados pelo CAPE na seleção de características. Pode-se ver na Figura 5.1 a relevância relativa de cada atributo dentro do conjunto de dados. As relevâncias (eixo vertical) estão relacionadas aos autovalores associados aos atributos (eixo horizontal). As barras em preto representam os atributos descartados pela aplicação sucessiva do CAPE e a linha tracejada separa os atributos mais relevantes (metade) dos menos relevantes (a outra metade). Pode-se ver que 4 dos 5 atributos descartados estão acima da linha tracejada, e ainda mais, o atributo 7, considerado como o mais relevante, é um deles. Porém, o que se pode esperar se tais atributos considerados simultaneamente redundantes e relevantes forem substituídos pelos seus correspondentes menos relevantes?

Para encontrar os atributos correspondentes menos relevantes é necessário inicialmente organizar o conjunto $\mathcal{A} = \{a_1, \ldots, a_P\}$, composto de todos os atributos do conjunto de dados, em dois subconjuntos. O primeiro é o subconjunto $\mathcal{D}$, cujos elementos são todos os atributos descartados pelo algoritmo CAPE, e o segundo é o subconjunto $\mathcal{N}$, composto pelos atributos não descartados. Em seguida, partindo do atributo mais relevante do subconjunto $\mathcal{D}$ e seguindo a ordem decrescente de suas relevâncias, executar o seguinte algoritmo:

Passo 1: Calcular os coeficientes de correlação entre o vetor representativo do atributo a_i escolhido dentre os elementos do conjunto $\mathcal{D}$ e todos os vetores representativos dos atributos do conjunto $\mathcal{N}$;

Passo 2: Escolher o atributo $a_j \in \mathcal{N}$ que tenha maior coeficiente de correlação com $a_i \in \mathcal{D}$;

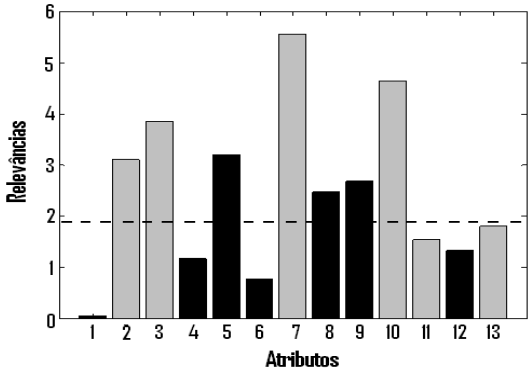

Figura 5.2: Panorama das relevâncias dos atributos do conjunto de dados *Wine* com a eliminação dos atributos (barras negras) pela associação CAPE-PCA.

Passo 3: Se a relevância de $a_j \in \mathcal{N}$ for menor que a relevância de $a_i \in \mathcal{D}$, faça

$$\begin{aligned} \mathcal{D} &\leftarrow \mathcal{D} - \{a_i\} + \{a_j\} \\ \mathcal{N} &\leftarrow \mathcal{N} - \{a_j\} + \{a_i\} \end{aligned} \tag{5.2}$$

Um alto índice de correlação sugere que os atributos envolvidos podem conter muita informação redundante. Assim, partindo do atributo descartado considerado o mais relevante (atributo 7) encontrou-se os pares redundantes $(7, 12)$, $(5, 13)$, $(2, 8)$, $(9, 7)$ e $(6, 7)$. Percebe-se que nos três primeiros pares o atributo redundante é coincidentemente menos relevante. Então, utilizou-se o algoritmo Levenberg-Marquardt para treinar novamente uma rede MLP com $Q = 9$ neurônios ocultos até atingir uma taxa de reconhecimento de 100%, mas com a eliminação dos atributos 6, 8, 9, 12 e 13. Em seguida, aplicou-se o CAPE. A poda resultou em: $Q = 4$, $N_c = 24$, $CR_{train} = 100\%$, $\varepsilon_{train} = 0,1920$ e $N_i = 6$. Houve a eliminação adicional dos atributos 2 e 5. Assim o conjunto $\mathcal{D}$ dos atributos descartados é $\mathcal{D} = \{2, 5, 6, 8, 9, 12, 13\}$. Entretanto, ainda é possível substituir os atributos 2 e 13 pelos seus pares menos relevantes 4 e 1, respectivamente. Assim, os atributos descartados são $\mathcal{D} = \{1, 4, 5, 6, 8, 9, 12\}$, como pode ser visto na Figura 5.2.

Com relação ao conjunto de dados *Dermatology*, a Figura 5.3 mostra o panorama dos atributos e suas estimativas de relevância para a condição final (16 atributos eliminados) alcançada na subseção 5.1. Pode-se ver que 9 atributos descartados são considerados como de baixa relevância, enquanto os outros 7 são considerados de alta relevância.

Conforme citado anteriormente, a aplicação do CAPE à Arquitetura 8 na Tabela 5.3 resultou na poda simultânea dos atributos 16 e 26. Entretanto, treinamentos posteriores, levando em consideração a eliminação adicional de ambos os atributos, não atingiram a

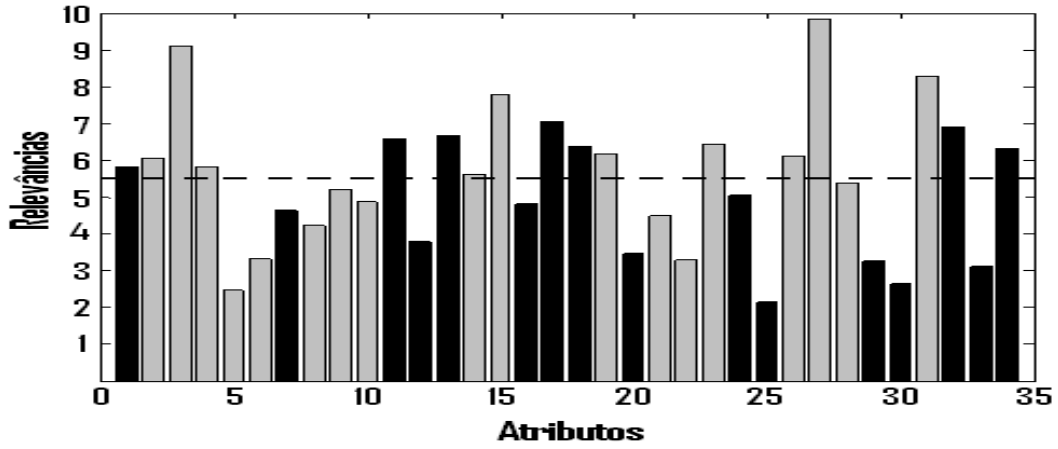

Figura 5.3: Panorama das relevâncias dos atributos do conjunto de dados *Dermatology* com a eliminação dos atributos (barras negras) pelo CAPE.

taxa de reconhecimento mínima aceitável. Testes adicionais sugeriram apenas a eliminação do atributo 16. Assim, com a aplicação da metodologia proposta nesta subseção, substituiu-se o atributo 26 pelo seu par redundante e menos relevante, o atributo 22, e implementou-se novo treinamento com o algoritmo Levenberg-Marquardt e apenas 17 atributos remanescentes. Como resultado, atingiu-se uma taxa de reconhecimento de 100%, o que confirma a representatividade do conjunto de dados por apenas 17 atributos.

Aplicações sucessivas de treinamento e poda com o CAPE levaram à eliminação adicional dos atributos 3, 10 e 23, perfazendo um total de 20 atributos descartados. A Figura 5.4 mostra o panorama final resultante da metodologia aplicada na seleção de características.

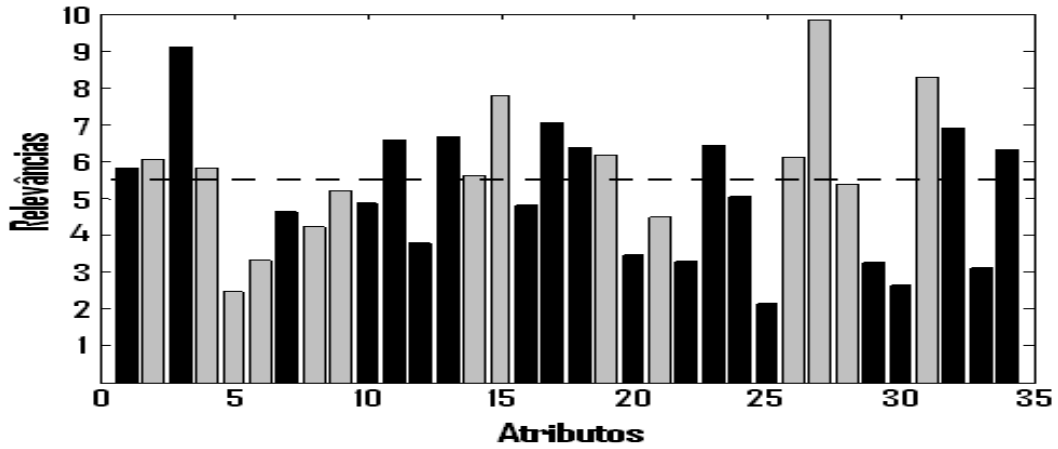

Figura 5.4: Panorama das relevâncias dos atributos do conjunto de dados *Dermatology* com a eliminação dos atributos (barras negras) pela associação CAPE-PCA.

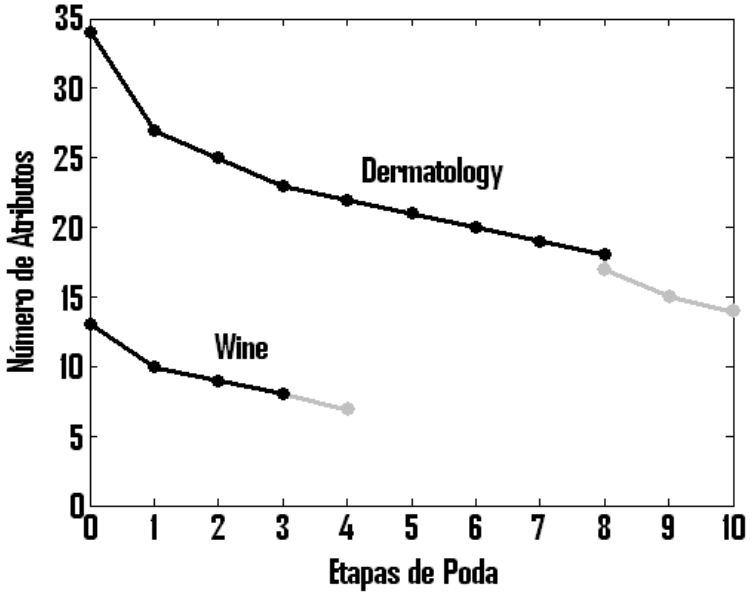

Figura 5.5: Número de atributos remanescentes ao final de cada etapa de seleção de características nos conjuntos de dados *Dermatology* e *Wine* pela associação CAPE-PCA.

Na Figura 5.5 pode-se ver a evolução na seleção de atributos nos conjuntos de dados *Wine* e *Dermatology*. Os pontos pretos estão relacionados ao número de atributos remanescentes ao final de cada etapa de seleção de características utilizando-se apenas o CAPE para seleção de características. Os pontos cinzas referem-se à aplicação da associação CAPE-PCA.

5.3 Conclusões

Neste capítulo foi apresentado o algoritmo CAPE como proposta metodológica para seleção de características. Resultados satisfatórios da aplicação do método foram apresentados a partir de simulações computacionais com conjuntos de dados bem conhecidos, *Wine* e *Dermatology*.

Em seguida, foi apresentado o algoritmo do PCA aplicado à rede dual como ferramenta auxiliar ao CAPE na estimativa de relevâncias de atributos. Com isso, pôde-se detectar redundância de atributos de forma mais agressiva, o que foi demonstrado com resultados de simulações computacionais.

No apêndice C pode-se ver outra aplicação do CAPE em seleção de características aplicado ao desenvolvimento de um detector neural de falhas de curto-circuito entre espiras estatóricas de motor de indução trifásico acionado por conversor de freqüência.

6 Conclusões e Perspectivas

Esta tese trata essencialmente de uma questão importante na aplicação de redes neurais: a capacidade de generalização.

O enfoque do estudo é a melhoria da capacidade de generalização do perceptron de múltiplas camadas, tendo em vista que a rede MLP com uma camada escondida é um aproximador universal e, por isso, tem atraído a atenção de pesquisadores e engenheiros de campo.

Para que se possa tirar vantagem dessa capacidade aproximativa da rede MLP é importante que se conheça seu potencial, bem como suas limitações e estratégias para treinamento, o que é tratado no capítulo 2. Sem este conhecimento, o usuário pode ter um longo e tedioso processo de busca pela solução desejada baseada no método de tentativa e erro, sem garantias de sucesso, o que é desencorajador.

O capítulo 3 apresenta um resumo de diversas técnicas e heurísticas disponíveis na literatura para orientar o projetista na escolha adequada de topologias, desenvolver ferramentas para análise do desempenho em função do custo computacional e criar estratégias para redução de estrutura com a manutenção do desempenho. Algumas das metodologias apresentadas são frutos da experimentação, mas outras são matematicamente refinadas e exigem conceitos matemáticos sólidos por parte do projetista.

Neste trabalho, é apresentada uma metodologia de poda de pesos sinápticos redundantes de uma rede MLP previamente treinado. Este é o enfoque do capítulo 4. Sem que haja qualquer suposição prévia, seja relativa ao processo de treinamento ou à forma da superfície de erro, o método dá subsídios para a poda de pesos sinápticos baseado no cálculo da correlação dos erros cometidos ou estimados por unidades de processamento de camadas sucessivas de uma rede MLP.

O desempenho do método proposto é avaliado (capítulo 4) em problemas de classificação de padrões e comparado aos resultados apresentados por métodos bem estabelecidos, com destaque para o OBS. O método proposto apresenta resultados equivalentes aos

apresentados pelo OBS, mas com a vantagem de ser de fácil implementação. É observado que a aplicação sucessiva da poda dos pesos sinápticos pode levar à completa eliminação de neurônios ocultos. Isto corrobora a idéia da aplicação de métodos de poda para a seleção de modelos neurais.

Como freqüentemente os projetistas se deparam com conjuntos de dados bastante limitados em termos de quantidade de exemplos, o uso de diferentes critérios para a poda, os quais também envolvem o conjunto de dados de teste, é experimentado. Os resultados mostram que o conjunto de teste pode desempenhar um papel importante na equalização das taxas de acerto nos conjuntos de treinamento, teste e validação, além de melhorar a capacidade de generalização da rede.

Outro resultado interessante está relacionado com o retreinamento de uma rede podada. Tomando a rede podada como ponto de partida para o retreinamento não parece gerar redes com maior capacidade de generalização. Entretanto, podas posteriores ao retreinamento podem ser efetivas na redução da topologia e manutenção da capacidade de generalização.

Como o método CAPE resulta da aplicação de uma técnica linear à rede dual realizou-se também simulações relativas à aplicação de PCA à rede dual para efeito de comparação. Verificou-se que, embora a aplicação do CAPE descarte mais conexões sinápticas, algumas vezes a aplicação do PCA à rede dual é mais eficiente no tocante a redução do custo computacional da rede podada.

No capítulo 5 o CAPE foi aplicado como ferramenta para seleção de características. Seus resultados foram bastante satisfatórios em simulações computacionais com conjuntos de dados bem conhecidos. A detecção de redundância foi especialmente ampliada com o uso combinado do CAPE com o PCA.

Esta pesquisa ainda pode ter os seguintes desdobramentos:

- Hassibi et al. (1993) demonstrou com seus experimentos que as redes podadas com o OBS sempre apresentam menores erros no conjunto de treinamento comparados com a aplicação de outros métodos de poda. Neste trabalho, o critério de poda utilizado foi basicamente a taxa de classificação da rede. Assim, verificou-se que, em muitas situações, mas não em todas, o OBS realmente mostrou-se superior neste quesito, mesmo quando as taxas de acerto no conjunto de treinamento promovidas por ambos os métodos são as mesmas. Em pesquisas posteriores pode-se investigar a amplitude da poda, com o erro quadrático médio usado como critério de poda, e

comparar seus resultados com os alcançados pelo OBS. Problemas de aproximação de funções ou identificação de sistemas parecem ideais para esta empreitada;

- Estender a aplicação do CAPE a redes do tipo RBF (*Radial Basis Function*);
- Desenvolver um critério de poda que leve em consideração as ativações dos neurônios da camada de saída com o objetivo de projetar classificadores com a imposição de margem mínima de separação entre classes (comparando com o *Support Vector Machine*;
- Realizar estudo comparativo entre o OBS e o CAPE em relação a robustez quanto ao tamanho do conjunto de treinamento. A aplicação do OBS depende do cálculo iterativo da matriz hessiana, o qual depende diretamente do número de padrões no conjunto de treinamento. A aplicação do CAPE é independente do número de padrões no conjunto de treinamento;
- É comum encontrar conjuntos de dados nos quais alguns de seus padrões apresentam atributos com valores faltantes. Quando o conjunto de dados é numeroso, muitas vezes o projetista descarta estes padrões. Mas quando o número de padrões disponíveis é pequeno deve-se tomar outra estratégia. Uma delas é atribuir ao elemento faltante o valor médio deste atributo, tomando para cálculo todo o conjunto de dados. Entretanto, acredita-se que uma rede MLP treinada com todo o conjunto de dados e subseqüentemente podada sob determinados critérios pode ser utilizada para realizar a recuperação de tais atributos. A idéia é apresentar sucessivamente o padrão com valor de atributo faltante à rede podada e permitir que a rede MLP dual ajuste sucessivamente o valor deste atributo até que haja convergência. Supostamente, sob convergência, o valor do atributo pode ser considerado como recuperado;
- Alguns projetistas se utilizam de índices, tais como o AIC, para avaliar o compromisso entre a representatividade do mapeamento entrada-saída e a complexidade de modelos neurais. Eles fornecem subsídios para o projetista escolher, dentre diversas topologias com diversos desempenhos diferentes, qual é a mais conveniente para a utilização final. Nestes índices, a complexidade do modelo geralmente está relacionada com o número de conexões sinápticas. Entretanto, no caso de redes podadas, o número de conexões sinápticas restantes (pesos diferentes de zero) difere do número de elementos das matrizes representativas do modelo neural, os quais são efetivamente utilizados no processamento da rede. Assim, trabalhos futuros podem relacionar o número de elementos das matrizes representativas do modelo, o desem-

penho do modelo e o valor médio das sensibilidades da função custo em relação aos pesos ($(\frac{dE}{dw})_{med}$) da rede, para a formação de um índice que avalie o compromisso entre representatividade e custo computacional.

- Implementar em *hardware* o detector neural projetado no apêndice C e estender a técnica para detecção de falhas também nas barras rotóricas do motor de indução.

APÊNDICE A – Implementação do OBS e do OBD

A.1 Cálculo da Inversa da Matriz Hessiana no OBS

Conforme visto na subseção 3.5.3, o método de poda OBS depende do cálculo dos elementos da diagonal principal da matriz inversa da Hessiana, Equação (3.45). Esta é uma matriz W x W, onde W é o número de pesos da rede. Para redes de pequeno porte esta matriz não apresentaria grandes dificuldades para a inversão. Entretanto, redes com topologias avantajadas são comuns, o que apresenta a primeira dificuldade pelo menos no que se refere a esforço computacional. Outra dificuldade é o fato de que, principalmente para redes de grande porte, a matriz Hessiana é mal-condicionada.

Hassibi et al. (1993) apresenta uma forma recursiva para o cálculo da inversa da matriz Hessiana de uma rede treinada com convergência. Considerando uma rede MLP com apenas um neurônio de saída, por simplificação, a equação do erro quadrático médio obtido com a rede sobre um conjunto com N dados de treinamento é dada por

$$E_{med} = \frac{1}{2N} \sum_{t=1}^{N} \left[d(t) - y^{(o)}(t) \right]^2, \tag{A.1}$$

onde $y^{(o)}(t) = F(\bar{\boldsymbol{\theta}}, x(t))$ é a função de mapeamento entrada-saída realizado pelo modelo ($\bar{\boldsymbol{\theta}}$). As derivadas primeira e segunda de E_{med} em relação a $\boldsymbol{\theta}$ são dadas por

$$\frac{dE_{med}}{d\boldsymbol{\theta}} = -\frac{1}{N} \sum_{t=1}^{N} \left(\frac{dF(\bar{\boldsymbol{\theta}}, x(t))}{d\boldsymbol{\theta}} \right) \left(d(t) - F(\bar{\boldsymbol{\theta}}, x(t)) \right) \tag{A.2}$$

e

$$\frac{d^2E_{med}}{d\boldsymbol{\theta}^2} = \frac{1}{N}\sum_{t=1}^{N}\left[\left(\frac{dF(\bar{\boldsymbol{\theta}},x(t))}{d\boldsymbol{\theta}}\right)\left(\frac{dF(\bar{\boldsymbol{\theta}},x(t))}{d\boldsymbol{\theta}}\right)^T - \frac{d^2F(\bar{\boldsymbol{\theta}},x(t))}{d\boldsymbol{\theta}^2}(d(t)-F(\bar{\boldsymbol{\theta}},x(t)))\right]. \tag{A.3}$$

Mas, considerando que no treinamento houve convergência para um mínimo global ou local da superfície de erro, a segunda parcela da Equação (A.3) pode ser desprezada em função dos baixos valores de $(d(t)-F(\bar{\boldsymbol{\theta}},x(t))$. Logo, a derivada segunda de E_{med} em relação a $\boldsymbol{\theta}$, ou seja, a matriz Hessiana pode ser aproximada por

$$H(\bar{\boldsymbol{\theta}}) = \frac{d^2E_{med}}{d\boldsymbol{\theta}^2} \cong \frac{1}{N}\sum_{t=1}^{N}\left(\frac{dF(\bar{\boldsymbol{\theta}},x(t))}{d\boldsymbol{\theta}}\right)\left(\frac{dF(\bar{\boldsymbol{\theta}},x(t))}{d\boldsymbol{\theta}}\right)^T. \tag{A.4}$$

Para simplificar a notação, um vetor W x 1 é definido como

$$\boldsymbol{\xi}(t) = \frac{1}{\sqrt{N}}\frac{dF(\bar{\boldsymbol{\theta}},x(t))}{d\boldsymbol{\theta}}, \tag{A.5}$$

onde os elementos do vetor $\boldsymbol{\xi}(t)$ referentes aos pesos da camada de saída e camada oculta são calculados respectivamente a partir de

$$\frac{dE_{med}}{d\theta_{ki}} = -y_i^{(h)}(t)\varphi_k^{'}(u_k^{(o)}(t))e_k(t) \tag{A.6}$$

e

$$\frac{dE_{med}}{d\theta_{ij}} = -x_j(t)\varphi_i^{'}(u_i^{(h)}(t))\sum_{k=1}^{M}\theta_{ki}\varphi_k^{'}(u_k^{(o)}(t))e_k(t). \tag{A.7}$$

Dessa forma pode-se aplicar a Equação (A.8) para a computação recursiva da inversa da Hessiana:

$$\mathbf{H}^{-1}(t) = \mathbf{H}^{-1}(t-1) - \frac{\mathbf{H}^{-1}(t-1)\boldsymbol{\xi}(t)\boldsymbol{\xi}^T(t)\mathbf{H}^{-1}(t-1)}{1+\boldsymbol{\xi}(t)\mathbf{H}^{-1}(t-1)\boldsymbol{\xi}^T(t)}. \tag{A.8}$$

A inicialização do algoritmo deve ser feita com $\mathbf{H}^{-1}(0) = \sigma^{-1}\mathbf{I}$ grande e a recursão deve ser realizada até que as N amostras sejam apresentadas. Hassibi et al. (1993) recomendam $10^{-8} \leq \sigma \leq 10^{-4}$.

A aplicação do OBS é dependente dos elementos da diagonal principal da matriz inversa da matriz Hessiana. Assim, o ajuste de σ, para a inicialização de $\mathbf{H}^{-1}(0)$, e o número

Tabela A.1: Influência do σ sobre a poda do OBS.

N	σ	Q	N_c	CR_{train}	ε_{train}	CR_{test}	ε_{test}
1	0,5	21	119	89,68	0,1910	86,96	0,1413
	0,1	21	113	89,68	0,1684	78,80	0,1940
	0,0001	20	107	89,68	0,1999	86,96	0,1680
	0,000001	22	113	89,68	0,1783	86,41	0,1438
5	0,5	21	113	89,68	0,1684	78,80	0,1940
	0,1	21	114	89,68	0,1736	86,96	0,1466
	0,0001	20	95	89,68	0,1775	82,07	0,1829
	0,000001	21	117	89,68	0,1662	84,24	0,1639
10	0,5	21	102	89,68	0,1707	82,61	0,1698
	0,1	22	123	89,68	0,1725	86,96	0,1428
	0,0001	19	101	89,68	0,2128	84,78	0,2104
	0,000001	20	112	89,68	0,1663	82,61	0,1756

de dados no conjunto de treinamento, responsável pelo número de iterações da recursão, passam a ser fatores influentes na qualidade da matriz inversa. A Tabela A.1 mostra os resultados obtidos com a aplicação do OBS à Arquitetura 1 da Tabela 4.8. Nesta, N representa o número de vezes que o conjunto de dados de treinamento é apresentado para o processamento da recursão da inversa da Hessiana. Para $N = 1$ a recursão é realizada com 126 iterações. Para $N = 10$, são realizadas 1260 iterações.

Embora não se tenha realizado extensivos experimentos, pode-se perceber que os resultados da poda sofrem a influência do parâmetro σ de inicialização da recursão e do número N de dados do conjunto de treinamento. Entretanto, não há indicação de qualquer tendência na qualidade da poda em função do σ ou de N.

Para investigar o efeito do σ sobre a poda de pesos foram realizados dois ensaios de inversão recursiva com inicializações diferentes. As Figuras A.1 e A.2 mostram a evolução da magnitude de 17 elementos da diagonal principal de uma matriz inversa em função do número de iterações durante o processo recursivo dado pela Equação (A.8). A matriz Hessiana em questão é referente a Arquitetura 4 ($N_c = 17$) do problema apresentado na subseção 4.3.2. Na Figura A.1 o valor de sigma é igual a 0,1 e na Figura A.2 é igual a 0,0001. Percebe-se perfis de convergência diferentes, mas os elementos da diagonal principal da inversa da matriz Hessiana correspondentes aos pesos podados (em negrito) pelo OBS convergem para os mais altos valores dentre todos, em ambas situações. Neste caso, pelo menos, o problema apresentou-se insensível à mudança do σ, mas este resultado não pode ser generalizado como pôde ser visto na Tabela A.1.

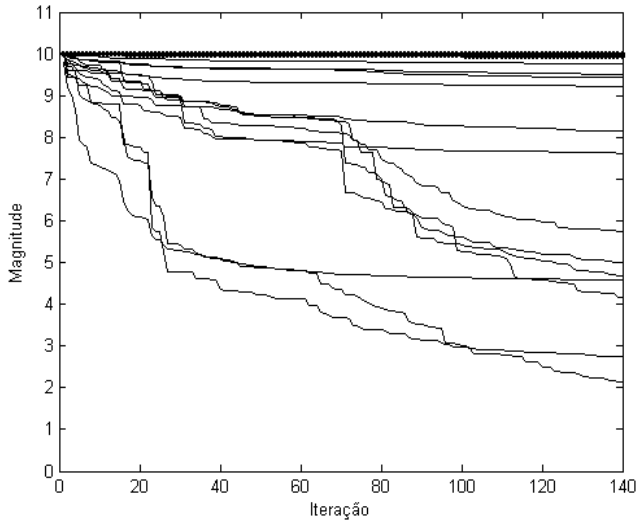

Figura A.1: Convergência dos elementos da diagonal principal durante a inversão recursiva de matriz Hessiana (σ=0,1).

A.2 Cálculo da Inversa da Matriz Hessiana no OBD

No caso da aplicação do OBD não há parâmetros a ajustar, pois o método não depende de recursões. A matriz Hessiana é simplificada para conter elementos diferentes de zero apenas na diagonal principal, o que simplifica sobremaneira a inversão. Os elementos da diagonal principal, referentes à derivada segunda da função custo em relação aos pesos da camada de saída e oculta, são calculados respectivamente por

$$\frac{d^2E_{med}}{d\theta_{ki}^2} = \frac{1}{N}\sum_{t=1}^{N}\left(y_i^{(h)}(t)\right)^2\left[\left(\varphi_k^{'}(u_k^{(o)}(t))\right)^2 - \varphi_k^{''}(u_k^{(o)}(t))e_k(t)\right] \tag{A.9}$$

e

$$\begin{aligned}\frac{d^2E_{med}}{d\theta_{ij}^2} = &+\frac{1}{N}\sum_{t=1}^{N}(x_j(t))^2\left[\left(\varphi_i^{'}(u_i^{(h)}(t))\right)^2\sum_{k=1}^{M}\theta_{ki}^2\left(\left(\varphi_k^{'}(u_k^{(o)}(t))\right)^2 - \varphi_k^{''}(u_k^{(o)}(t))e_k(t)\right)\right] + \\ &-\frac{1}{N}\sum_{t=1}^{N}(x_j(t))^2\left[\varphi_i^{''}(u_i^{(h)}(t))\sum_{k=1}^{M}\theta_{ki}\varphi_k^{'}(u_k^{(o)}(t))e_k(t)\right].\end{aligned} \tag{A.10}$$

Agora, resta apenas aplicar os valores obtidos com as Equações (A.9) e (A.10) diretamente em $S_i = \frac{1}{2}\frac{d^2E_{med}}{d\theta_i^2}.\theta_i^2$ para calcular as sensibilidades.

O cálculo analítico da matriz Hessiana de MLPs com mais de uma camada oculta é

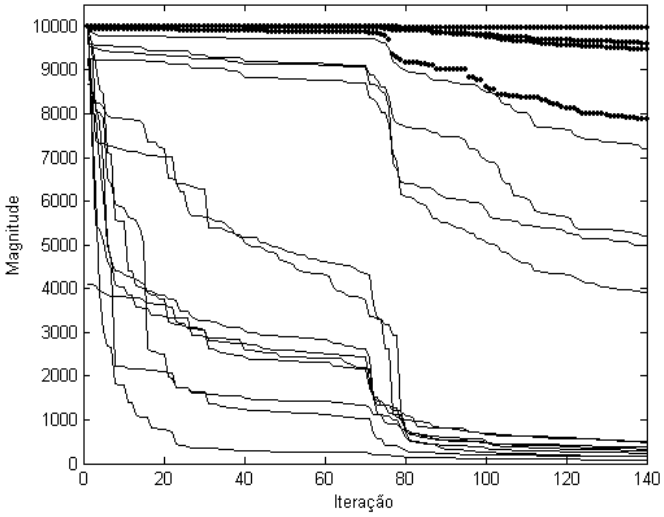

Figura A.2: Convergência dos elementos da diagonal principal durante a inversão recursiva de matriz Hessiana (σ=0,0001).

Tabela A.2: Comparação entre o OBS e o OBD.

Dados	Método	Q	N_c	CR_{train}	ε_{train}	CR_{test}	ε_{test}
	rede treinada	9	75	99,05	0,0127	93,33	0,0858
Iris	OBD1	2	11	99,05	0,2868	93,33	0,3793
	OBD2	4	15	99,05	0,2702	93,33	0,3480
	OBS	2	11	99,05	0,2868	93,33	0,3793
	rede treinada	24	243	89,68	0,1132	87,50	0,1274
Coluna	OBD1	21	121	89.68	0,1729	84,24	0,1670
	OBD2	21	122	89.68	0,1887	79,89	0,1671
	OBS	20	107	89.68	0,1999	86,96	0,1680
	rede treinada	9	156	100	0.0007	97,47	0,0403
Wine	OBD1	3	24	100	0,1572	93,67	0,2308
	OBD2	3	25	100	0,2664	93,67	0,3452
	OBS	5	27	100	0,2702	97,47	0,2644

tratado em Bishop (1992).

A Tabela A.2 mostra resultados obtidos com a aplicação do OBS e do OBD a conjuntos de dados tratados nas subseções 4.3.2, 4.3.3 e 5, respectivamente. As denominações OBD1 e OBD2 referem-se a duas formas de calcular os elementos da diagonal da matriz Hessiana. Em OBD1, os elementos são obtidos a partir do produto externo do vetor $\xi(t)$ pelo seu transposto. Já em OBD2, o cálculo é realizado a partir das Equações (A.9) e (A.10). Percebe-se que os erros nos conjuntos de treinamento alcançados pelas topologias podadas pelo OBD são em geral menores que os alcançados pelo OBS. Entretanto, os acertos nos conjuntos de teste têm se mostrado melhores com o OBS.

APÊNDICE B – Detecção de falhas em MITs

Estudos relativamente recentes mostraram que aproximadamente 50% de todas as cargas elétricas na indústria brasileira moderna são motores de indução (SANTOS et al., 2001). Isto se dá pois o motor de indução trifásico gaiola de esquilo apresenta grande robustez e versatilidade (BONNETT; SOUKUP, 1992), principalmente com a aplicação de conversores de freqüência.

Apesar das técnicas de projeto e construção dos motores de indução trifásicos terem atingido alto grau tecnológico, a máquina em si tem suas limitações, as quais, se excedidas, resultarão em falhas prematuras no estator ou no rotor (LI et al., 1997). Logo, a aplicação de técnicas de detecção prematura de falhas pode reduzir significantemente custos de manutenção e de produção industrial (HODGE; AUSTIN, 2004). Mas para isto, é necessário que se reconheça os sintomas característicos de cada tipo de falha específica.

A seguir são caracterizadas as falhas mais comuns nos motores de indução trifásicos do tipo gaiola de esquilo.

B.1 Falhas em Rolamentos

O desempenho de motores elétricos está intimamente relacionada com a qualidade da montagem de seus rolamentos. Além disso, de forma crescente as falhas em motores estão geralmente relacionadas com falhas nos rolamentos. Tais falhas se manifestam como falhas de assimetria rotórica e geralmente são consideradas como pertencentes à categoria de falhas por excentricidade. Como, no entanto, entre 40% e 50% de todas as falhas ocorridas em motores estão relacionadas com os rolamentos (NANDI et al., 2005), estas merecem um estudo a parte.

A maioria das máquinas elétricas usa rolamentos de esferas ou rolos, os quais, mesmo sob condições normais de operação, com carga balanceada e bom alinhamento, podem sofrer falha por fadiga. Dentre as causas internas à máquina, os rolamentos podem sofrer

estresse devido a vibrações, excentricidade inerente e circulação de corrente pelos seus corpos (devido ao uso de conversores eletrônicos). Os rolamentos também podem ser atingidos por causas externas, tais como: a) Contaminação por partículas sólidas, as quais podem provocar arranhões ou mesmos buracos nas pistas ou elementos rolantes, e corrosão por ação de água, ácidos etc; b) Lubrificação imprópria, considerando aqui tanto o excesso quanto a falta, cujos efeitos são o sobreaquecimento e a abrasão; c) Instalação inadequada. Como consequência, frequentemente ocorre o crescimento nos níveis de vibração e ruído.

Geralmente os rolamentos consistem de dois anéis concêntricos, chamados de pista interna e pista externa, com um grupo de elementos rolantes correndo entre estas duas pistas. Tipicamente, estes elementos rolantes (esferas, rolos etc) são guiados por uma gaiola que assegura o espaçamento uniforme e evita o contato mútuo entre os elementos rolantes.

Quando ocorrem falhas em um rolamento, impulsos periódicos são gerados como conseqüência da passagem de elementos rolantes pelos pontos de falha nas pistas interna e externa. Como conseqüência, o rolamento vibra em sua freqüência natural modulada pela freqüência proveniente da falha (HYUN; NAM, 1995). Com isso, sinais provenientes de sensores de vibração podem ser medidos e comparados com medidas de referência com o objetivo de interpretar as condições de operação dos rolamentos.

Logo, faz-se necessário o estudo da dinâmica dos rolamentos, a qual é descrita a partir de cinco movimentos básicos, os quais apresentam freqüências correspondentes. Estas cinco freqüências são denominadas de freqüência rotacional do eixo (F_s), freqüência fudamental da gaiola (F_c), freqüência de passagem do elemento rolante pela pista interna (F_{bpi}), freqüência de passagem do elemento rolante pela pista externa (F_{bpo}) e freqüência de rotação do elemento rolante (F_b) (LI et al., 2000). Estas freqüências estão ilustradas na Figura B.1.

A velocidade do eixo é fundamental para a determinação dos movimentos em um rolamento. Entretanto, é mais conveniente representar as velocidades lineares por suas correspondentes freqüências rotacionais. Assim, todas as outras freqüências estão relacionadas com F_s.

A freqüência fudamental da gaiola pode ser derivada a partir da velocidade linear de algum ponto da gaiola (V_c), como sendo a média das velocidades lineares das pistas interna e externa

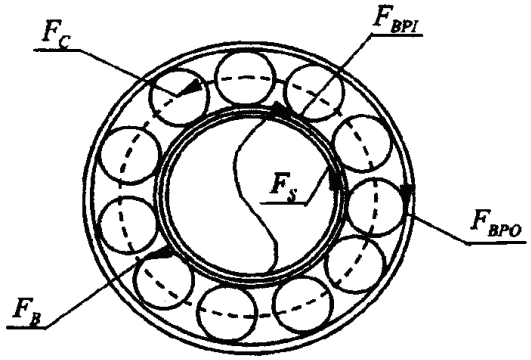

Figura B.1: Freqüências básicas em rolamentos.

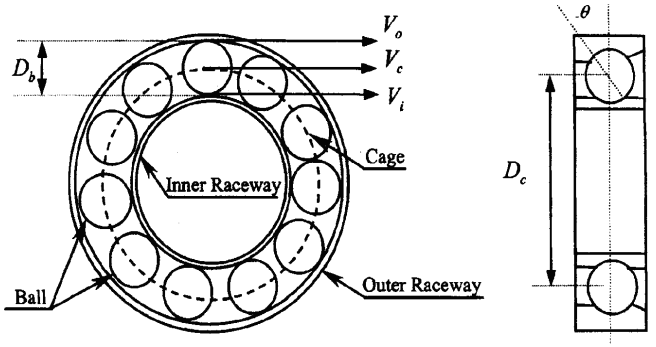

Figura B.2: Estrutura de um rolamento com esferas e definição de cada variável.

$$F_C = \frac{V_C}{R_C} = \frac{V_i + V_0}{D_c}. \tag{B.1}$$

Uma consideração realista é que $V_i = V_s$ e $V_o = 0$ (Figura B.2). Então, a velocidade linear pode ser expressa pela sua respectiva freqüência rotacional

$$F_C = \frac{F_s}{2}\left(1 - \frac{D_B cos\theta}{D_C}\right). \tag{B.2}$$

As freqüências de passagem do elemento rolante pela pista interna (F_{bpi}), de passagem do elemento rolante pela pista externa (F_{bpo}) e de rotação do elemento rolante (F_b) são dadas por

$$F_{bpi} = \frac{N_B}{2} F_s \left(1 + \frac{D_B cos\theta}{D_C}\right), \tag{B.3}$$

$$F_{bpo} = \frac{N_B}{2} F_s \left(1 - \frac{D_B cos\theta}{D_C}\right) \tag{B.4}$$

e

$$F_b = \frac{D_C}{2D_B} F_s \left(1 - \frac{D_B^2 cos^2\theta}{D_C^2}\right). \tag{B.5}$$

Estudos realizados no domínio da freqüência mostram que quando ocorrem defeitos em rolamentos, os sinais de vibração apresentam algumas das freqüências acima.

Considerando um único defeito na pista interna, o sinal de vibração conterá F_{bpi}. Da mesma forma, se um único defeito ocorre na pista externa, o sinal de vibração conterá F_{bpo}. Porém, se a área do defeito for grande, o sinal de vibração conterá harmônicos, podendo servir de indicação do quão severa é a falha. Para falha em um elemento rolante o sinal de vibração conterá $2F_B$, tendo em vista que o ponto de falha entrará em contato com ambas as pistas. Em muitos casos, esta freqüência estará combinada com F_{bpi} e F_{bpo}, resultando em um espectro mais complicado. Caso a área de defeito no elemento rolante seja grande, a freqüência natural do sistema também estará excitada e modulada com duas vezes a freqüência rotacional do elemento rolante (LI et al., 2000).

B.2 Falhas por Excentricidade Estática e Dinâmica

Existem dois tipos de excentricidade rotórica. A excentricidade estática, onde a posição na qual ocorre o menor comprimento radial de entreferro é fixa no espaço, e a excentricidade dinâmica, onde a posição de mínimo comprimento radial de entreferro gira com o rotor.

Pode ser visto na Figura B.3 que $C1$ é o centro do rotor, e $C2$ é o centro do estator. Com a excentricidade estática, $C1$ é o centro de rotação, e com excentricidade dinâmica, $C2$ é o centro de rotação. Com a combinação das excentricidades estática e dinâmica, o centro de rotação pode ser qualquer ponto entre $C1$ e $C2$.

A excentricidade estática pode ser causada por ovalização do núcleo estatórico ou pelo posicionamento incorreto do rotor ou estator durante o processo de montagem. Já a excentricidade dinâmica pode ser causada por empenamento do eixo, desalinhamento ou desgaste de rolamentos, ressonância mecânica em velocidades críticas etc.

Algum nível de excentricidade estática pode ser encontrado em máquinas recém produzidas devido ao processo de fabricação ou método de montagem. No entanto, apesar de uma excentricidade de até 10% ser admissível para motores novos (NANDI et al., 2005), os fabricantes de motores normalmente mantêm a excentricidade total em níveis mais

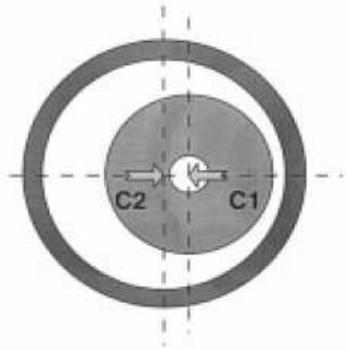

Figura B.3: Excentricidade em máquinas elétricas.

baixos para reduzir ruídos, vibrações e forças magnéticas radiais. Os fabricantes também sugerem que excentricidades da ordem de 20% sejam consideradas inaceitáveis e que um nível de 50% seja considerado sério o bastante para que haja a imediata remoção de serviço (BARBOUR; THOMSON, 1997).

As forças magnéticas a que se faz menção no parágrafo anterior puxam o rotor do centro do núcleo estatórico na direção da posição espacial de menor entreferro. No caso de excentricidade dinâmica, ocorrem forças magnéticas desbalanceadas que giram na velocidade do rotor (DORRELL et al., 1997). Estas forças podem, com o uso do motor, ocasionar empenamentos de eixo e desgaste de rolamentos, levando a choques mecânicos danosos entre rotor e estator.

Os efeitos das excentricidades estática e dinâmica sobre a composição espectral da corrente estatórica podem ser avaliados a partir de uma aproximação clássica das ondas de força magneto-motriz e de permeância. Em (DORRELL et al., 1997) é mostrado que a variação do fluxo devido a excentricidades gera tensões tanto nas barras do rotor quanto no bobinamento estatórico. Dessa forma, correntes com freqüências dadas por

$$f_{ec} = \left(f \pm f_r\right) \tag{B.6}$$

passam a figurar no espectro harmônico da corrente estatórica.

Barbour & Thomson (1997) fazem uma investigação com elementos finitos, a qual revelou que o projeto das ranhuras rotóricas tem considerável efeito sobre a magnitude das componentes de corrente, especialmente quando uma excentricidade estática está presente. Ainda mais, o projeto da ranhura também afeta o tamanho do crescimento da magnitude dos componentes de corrente com o crescimento do nível de excentricidade estática. Os

mesmos propõem uma Equação (B.7), a qual é mais geral para a composição harmônica da corrente estatórica em função da variação da permeância ao redor do entreferro devido as ranhuras rotóricas, saturação e excentricidade.

$$f_{ec} = f_1 \left[\left(R \pm n_d \right) \left(\frac{(1-s)}{p} \right) \pm n_{ws} \right] \tag{B.7}$$

onde

f_{ec} - componente de freqüência devido à excentricidade

f_1 - freqüência da rede de alimentação

R - número de ranhuras rotóricas

n_d - constante com valor "0" para excentricidade estática
e "1" para dinâmica

s - escorregamento

p - pares de pólos

n_{ws} - constante representativa dos harmônicos temporais da força magneto-motriz
que pode assumir valores 1,3,5,7,...

Porém, para que a componente realmente exista no espectro de freqüência da corrente estatórica é necessário que o número de pólos dado pela Equação (B.8) seja compatível com o número de pólos estatóricos de projeto (THOMSON; BARBOUR, 1998).

$$m = \left(R \pm S \pm n_s \pm n_d \pm 2n_{sa}p \pm n_{\theta s}p \right) \tag{B.8}$$

onde

m - número de pares de pólos da ondas de fluxo

R - número de ranhuras rotóricas

S - número de ranhuras estatóricas

n_s - constante de excentricidade estática

n_d - constante de excentricidade dinâmica

n_{sa} - constante de saturação

$n_{\theta s}$ - constante de harmônica espacial rotórica

A Equação (B.7) pode ser utilizada para detectar excentricidade, porém a determinação do que é excentricidade estática ou dinâmica ainda é inconclusiva, tendo em vista que alto nível de excentricidade estática também produz componentes de excentricidade dinâmica (THOMSON et al., 1999). Além disto é importante saber que a magnitude das componentes harmônicas introduzidas pela excentricidade diminui com a entrada de carga (BARBOUR; THOMSON, 1997).

A detecção das componentes dadas pela Equação (B.7) no espectro de corrente deve ser precedida por uma redução da magnitude da componente principal da fonte de alimentação através de um filtro passa alta. Em seguida, devem ser implementadas a aquisição e análise do sinal, pois podem ocorrer problemas de interferência devido a outros sinais provenientes de cargas com pulsação mecânica, tais como compressores, máquinas de pulverização e acionamentos com caixas de engrenagens (THOMSON et al., 1999).

Nandi et al. (2002) propõem uma modificação na Equação (B.6) com a inclusão de um inteiro "k", resultando na Equação (B.9). Através de simulações computacionais justifica a sua funcionalidade. Afirma que a componente de freqüência dominante na Equação (B.9) ($k = 1$) fornece uma boa medida de excentricidade combinada, independentemente das combinações de número de pares de pólos e número de ranhuras.

$$f_{ec} = \left| f \pm k f_r \right| \tag{B.9}$$

Nandi et al. (2002) também mostram através de simulação computacional que as freqüências dadas pela Equação (B.9) apresentam influência sobre as componentes de alta freqüência . Assim, incluem o inteiro "k" na Equação (B.7), agora representada por

$$f_{ec} = f_1 \left[\left(kR \pm n_d \right) \left(\frac{(1-s)}{p} \right) \pm n_{ws} \right]. \tag{B.10}$$

A investigação da influência de excentricidades sobre os componentes de alta freqüência levou a conclusão de que, para algumas máquinas, tais componentes não são facilmente detectáveis (NANDI et al., 2002). Assim, sempre que possível, a recomendação é monitorar as característica apresentadas pelos componentes de baixa e alta freqüência para propósitos de diagnose de falha por excentricidade

Cardoso & Saraiva (1993) utilizam um método não invasivo de detecção de falha por excentricidade baseado no monitoramento do vetor de correntes estatóricas de Park. Através da observação da mudança de padrão do vetor de correntes, é possível, inclusive,

obter informação qualitativa do grau de severidade da falha.

B.3 Falhas Devido Abertura de Barras ou Rachadura de Anéis

De 5% a 10% das falhas em motores de indução são devido à abertura de barras ou rachadura de anéis na gaiola de esquilo (NANDI et al., 2005). Normalmente a falha é iniciada nas conexões entre a barra e os anéis de curto-circuito (MULLER; LANDY, 2003). Estas falhas produzem concentração de linhas de campo magnético no entreferro na vizinhança do ponto de falha. Isto ocorre devido a interrupção das correntes induzidas nas barras defeituosas e a consequente ausência de força magneto-motriz produzida pelas mesmas. Como consequência se tem um aumento na oscilação de conjugado e redução do desempenho mecânico.

Elkasabgy et al. (1992) modelam a seção transversal de um motor de indução com a técnica de elementos finitos e obteve uma representação detalhada da distribuição típica do campo magnético na máquina com e sem abertura de barras.

Segundo Kliman et al. (1988), os indicadores mais comuns da abertura de barras num motor de indução com gaiola de esquilo são vibração excessiva, ruído e centelhamento durante a partida do motor. Tais sintomas são perceptíveis quando a falha já se estendeu a várias barras. Por isso, é importante o desenvolvimento de métodos de detecção precoce de falhas na gaiola, antes que os danos apresentem extensão catastrófica.

As correntes do circuito rotórico sob falha na gaiola produzem duas forças magneto-motrizes girantes, as quais geram fluxos que giram nas freqüências $+sf$ e $-sf$ respectivamente em relação a um observador no rotor. Tais fluxos induzem tensões e correntes estatóricas de freqüências f e $(1-2s)f$, onde f é a freqüência fundamental da rede de alimentação. A interação entre a componente de freqüência mais baixa com a componente de freqüência fundamental dá origem a uma oscilação de conjugado, o que pode levar a uma oscilação de velocidade com conseqüente elevação de ruído e vibração. Estas próprias oscilações de conjugado e de velocidade rotórica podem ser usadas para a detecção da abertura de barras e/ou seções dos anéis de curto-circuito da gaiola (MANOLAS; TEGOPOULOS, 1999).

O rotor operará com "ripple" de conjugado na freqüência $2sf$ sobreposto ao conjugado fundamental, o que levará a um "ripple" de velocidade sobreposto à freqüência mecânica fundamental de $(1-s)f$. Durante a excursão positiva do "ripple" haverá a indução de

tensões e correntes de freqüências superiores a $+sf$ e $-sf$ no rotor, o que dará origem a tensões e correntes induzidas no estator de freqüências superiores a f.

Segundo Filippetti et al. (1998) a perturbação no espectro harmônico da corrente devido a assimetrias no circuito rotórico é representada por

$$f_b = \left(1 \pm 2ks\right) f \tag{B.11}$$

onde k pode assumir valores inteiros positivos.

Restringindo a Equação (B.11) para a condição em que k é igual a 1, tem-se que a freqüência inferior é um reflexo direto da abertura de barras na gaiola enquanto a freqüência superior é uma conseqüência de oscilações de velocidade.

Alguns pesquisadores têm usado a análise espectral de freqüência da corrente estatórica de motores de indução para a detecção de falhas na gaiola de esquilo. A amplitude da componente de freqüência $(1-2s)f$ pode ser utilizada para quantificação da extensão da falha na gaiola. Porém, Filippetti et al. (1998) recomenda que se utilize a soma das amplitudes das componentes de $(1-2s)f$ e $(1+2s)f$ na presença de oscilação de velocidade, pois o momento de inércia da associação motor-carga tem grande influência no espectro harmônico da corrente estatórica. Quando pequeno, permite grandes oscilações de velocidade, o que reduz a amplitude de $(1-2s)f$ e aumenta a amplitude de $(1+2s)f$. Quando grande ocorre o inverso. Bellini et al. (2001) chama atenção para o fato de que a onda espacial de densidade de fluxo saturado de terceira ordem no entreferro também produz a componente estatórica de $(1+2s)f$, o que dificulta a detecção de falha nas barras.

Segundo Kliman et al. (1988), as assimetrias decorrentes do processo de fabricação podem produzir falsas indicações de falhas em barras se o método de detecção estiver focado na observação das freqüências em torno da freqüência fundamental do circuito estatórico. Segundo eles, as amplitudes das harmônicas de mais alta freqüência devido a abertura de barras são mais fácil distinção das harmônicas devido a assimetrias.

Milimonfared et al. (1999) implementam um método de detecção de falha na gaiola de motores de indução trifásicos baseado na análise do espectro harmônico das tensões induzidas no bobinamento estatórico pelo fluxo rotórico após a desconexão da rede de alimentação. Na ausência de falha a tensão induzida é predominantemente senoidal. Caso contrário a tensão induzida apresenta distorções em relação a uma senóide. Busca-se com isto evitar que os efeitos indesejáveis de saturação, desbalanceamento de tensões e inserção

de harmônicos pela fonte venham a dificultar a detecção da falha. Contudo, devido a inerentes assimetrias na máquina, existem ainda algumas componentes harmônicas, mesmo numa máquina em perfeitas condições de operação. Percebeu-se que embora o número de barras com falhas não tenha muito efeito sobre a magnitude das componentes harmônicas, é possível distinguir uma máquina com falha de outra em perfeitas condições de operação.

Meshgin-Kelk et al. (2004) utilizam um modelo tridimensional para avaliar a influência do fluxo produzido pelas correntes entre barras. Este fluxo percorre o eixo, as tampas frontal e traseira, os núcleos rotóricos e estatóricos e os dentes rotóricos e estatóricos do motor. Isto sugere que bobinas instaladas nas tampas do motor e que envolvam o eixo podem conter tensões induzidas devido aos fluxos axiais produzidos pelas correntes rotóricas de freqüências sf, $3sf$, $5sf$ ... . Os pesquisadores verificaram também que estas tensões induzidas são maiores nas bobinas mais próximas da falha.

B.4 Falhas no Bobinamento Estatórico

Essas falhas geralmente estão relacionadas com deficiências apresentadas pelos materiais isolantes. Elas são comumente conhecidas como falhas fase-terra ou fase-fase. Em torno de 30% a 40% de todas as falhas no motor de indução são reputadas a esta categoria (NANDI et al., 2005).

A isolação do bobinamento estatórico pode sofrer *stress* devido a diversas razões, dentre elas as pricipais são: alta temperatura do bobinamento ou núcleo estatóricos; folga na laminação do núcleo magnético estatórico, perda da proteção das conexões de bobinas; contaminação provocada por óleo, umidade ou sujeira; descargas elétricas; e vazamentos no sistema de arrefecimento, quando é o caso (NANDI et al., 2005). Como consequência, pode se dá início um processo de falha. Este é geralmente iniciado com uma falha de alta impedância (da ordem de vários kilo-ohms) entre espiras na mesma fase, entre fases ou para a terra (NATARANJAN, 1989).

A corrente de falta, a qual pode atingir a ordem de duas vezes a corrente de rotor travado, causa severo aquecimento localizado, provocando um rápido alastramento da falha no bobinamento (TALLAM et al., 2003). Falha no isolamento entre bobinamento e terra pode causar uma grande corrente para a terra, a qual resulta em danos irreversíveis ao núcleo da máquina. Se a falta for detectada no seu primeiro estágio de desenvolvimento, a máquina ainda poderá ser reutilizada após o rebobinamento do estator. Entretanto, o

tempo para a evolução de um curto-circuito entre espiras até um curto-circuito fase-fase ou fase-terra depende de muitas variáveis e ainda é uma questão difícil de se responder. Entretanto, a opinião geral dos fabricantes e usuários é que este tempo de evolução da falha em motores de baixa tensão é longo em relação às falhas ocorridas em motores de alta tensão (THOMSON, 2001).

Nataranjan (1989) descreve um método para detectar falhas iniciais em bobinamento estatóriocos de motores ligados na configuração estrela com neutro aterrado. Ele forçou a passagem dos três cabos das fases de alimentação do motor pela janela de um transformador de corrente. Sob simetria da rede elétrica e condições operacionais normais do motor, o fluxo magnético resultante no núcleo do transformador é zero e a corrente secundária também. Entretanto, quando da ocorrência de falha, o desbalanceamento das correntes no primário do transformador de corrente provoca a circulação de corrente secundária devido a componente de sequência zero.

Em Cash et al. (1997) os pesquisadores examinam desvios no valor médio quadrático (RMS) da soma das tensões fase-neutro ($Vsum$). Considerando apenas a componente fundamental, $Vsum$ numa máquina balanceada com a alimentação a três condutores é zero. Contudo, na presença de falha no isolamento, ocorrerá desbalanceamento e, então, a soma das três tensões fase-neutro será diferente de zero. Este método apresenta imunidade à variação de carga, desde que a carga afeta cada impedância de fase igualmente. O método também é imune a desbalanceamentos na fonte de tensão.

Agora, considerando a teoria de componentes simétricas, é sabido que o bobinamento trifásico do motor idealmente pode ser visto como um sistema simétrico, no qual apenas as correntes de sequência positiva circulam. Sob falha no isolamento, esta simetria é perdida e passam a circular correntes de sequência negativa. Isto sugere que a corrente de sequência negativa pode ser usada como indicador de falha no isolamento. Mas, em condições de campo, a corrente de sequência negativa pode ser observada em motores em perfeitas condições de uso devido a suas assimetrias inerentes e desbalanceamentos da fonte de alimentação.

No método proposto em Tallam et al. (2000), uma rede neural é usada para aprender o modelo da máquina (estimar o fasor de corrente de sequência negativa) sob condições de operação normal, incluindo os efeitos de desbalanceamento da fonte de alimentação e assimetrias inerentes ao motor. A magnitude da diferença entre o valor medido e o estimado para a componente de sequência negativa da corrente de linha é a medida da severidade da falha entre espiras.

O comportamento de um motor de indução em perfeitas condições operativas alimentado por tensões desbalanceadas também pode ser analisado pelo exame do seu circuito equivalente de sequência negativa. A componente variável da impedância deste circuito é dada por $R_2 = -(1-s)R_r/(2-s)$, a qual não é muito sensível ao escorregamento (s) do motor, tendo em vista que o motor opera com baixo escorregamento ($s < 0,03$). Sottile & Kohler (1993) observaram esta relativa invariabilidade em motores sob uma larga faixa de desbalanceamento da fonte de alimentação e condições de carga. Contudo, durante estágios iniciais do desenvolvimento de falhas no bobinamento, a simetria é perdida e a impedância de sequência negativa varia significantemente. Consequentemente, falhas podem ser detectadas pelo desvio em relação ao caso simétrico.

Penman et al. (1994) propõem um método para detectar falhas no bobinamento estatórico baseado na observação de determinadas harmônicas de tensão induzidas num bobinamento especialmente instalado concentricamente ao eixo do motor. Estas componentes harmônicas de tensão são induzidas devido ao fluxo axial da máquina. A idéia é que, na prática as assimetrias existem no estator e no rotor devido a pequenas imperfeições na geometria do bobinamento, assim como falta de uniformidade presente nos materiais de construção do motor. O efeito conjunto destes desvios da máquina ideal é o surgimento de um pequeno fluxo de acoplamento axial, porém mensurável. Então, como uma falha no bobinamento representa uma grande assimetria, o efeito desta falha deve salientar o fluxo axial.

A expressão geral das componentes harmônicas devido a falhas no bobinamento estatórico foi matematicamente derivada como

$$f_h = [k \pm n(1-s)/p]f_1 \tag{B.12}$$

onde p é o número de pares de pólos, s é o escorregamento, n é a ordem dos fluxos harmônicos espaciais, k é a ordem das harmônicas temporais da fonte de alimentação e f_1 é a freqüência fundamental da fonte de tensão. Apenas as harmônicas de baixa ordem são significantes. Os elementos chave para esta expressão ocorrem para $k = 1, 3$ e $n = 1, 2, 3, ..., (2p-1)$.

Embora a Equação (B.12) tenha sido desenvolvida para mostrar as componentes de freqüência no fluxo axial do eixo do motor, ela pode ser usada como referência para detectar falhas no bobinamento do motor pela observação do espectro de freqüência da corrente estatórica, pois esse fluxo também atinge as bobinas do estator.

Thomson (2001) se baseia em resultados experimentais para provar que a análise do espectro de freqüência da corrente do motor pode fornecer diagnóstico de falha por curto-circuito entre espiras. Foi observado que todos os espectros de freqüência das correntes de linha mostram o mesmo padrão, e então, um único transformador de corrente, instalado em qualquer das fases, é necessário para a detecção. Ele também afirma que a influência da carga sobre a magnitude das componentes de freqüência, dada pela Equação (B.12), é negligenciável diante da variação que ocorre devido ao curto-circuito de apenas uma espira. Ou seja, sob quaisquer condições de carga o efeito do curto-circuito entre espiras domina.

A análise apresentada em Stavrou et al. (2001) determina a escolha correta das componentes hamônicas a serem monitoradas. A partir de experimentos ficou evidente que algumas componentes de freqüência foram redundantes. Entretanto, a recomendação dos pesquisadores é que cada caso deve ser analisado particularmente.

Em Joksimovié & Penman (2000) é mostrado que, devido a própria natureza da gaiola do rotor, nenhuma componente nova no espectro de freqüência da corrente do motor aparece como consequência da falha no bobinamento estatórico. Na realidade observou-se que ocorre apenas o crescimento de algumas componentes já existentes. Foi demonstrado que sob condições de falha ocorre um crescimento significante nas componentes harmônicas devido as ranhuras rotóricas. Estas componentes de freqüência também são sensíveis a algumas outras condições anormais de funcionamento do motor, tais como desbalanceamento das tensões da fonte de alimentação e excentricidade estática. Então, estas componentes não podem ser usadas como um sinal específico da ocorrência de curto-circuito entre espiras no bobinamento estatórico.

Nandi & Toliyat (2000) propõem monitorar determinadas harmônicas relacionadas com as ranhuras rotóricas na tensão terminal da máquina após o desligamento da mesma. Na ausência da fonte de suprimento, harmônicas temporais devido a desbalanceamentos de tensões não influenciam na medição, exceto nas condições iniciais. Resultados satisfatórios foram obtidos a partir de simulações e experimentação, onde se obteve detecção com apenas $1,5\%$ $(5/324)$ de espiras em curto-circuito.

APÊNDICE C – Detecção de Curto-Circuito Entre Espiras Estatóricas de Motor de Indução Acionado por Conversor de Freqüência

A vida útil de um motor de indução trifásico está intimamente relacionada com a integridade do isolamento do bobinamento estatórico. No caso de motores acionados pela rede elétrica convencional em baixa tensão, as falhas no isolamento do bobinamento estatórico correspondem a algo em torno de 30% a 40% de todas as falhas no motor de indução (NANDI et al., 2005).

O desenvolvimento e a disseminação dos conversores de freqüência equipados com IGBTs (*Insulated Gated Bipolar Transistors*) têem sido acompanhados pelo crescimento da incidência de falhas no isolamento estatórico de motores de indução trifásicos (BELL; SUNG, 1997). Estes conversores utilizam a modulação por largura de pulso senoidal (PWM senoidal) para obter freqüência e tensão eficaz variáveis no acionamento dos motores em velocidades variáveis. As tensões, entretanto, são chaveadas a partir de freqüências de onda portadora que comumente atingem 20kHz. Sob tais condições, o isolamento experimenta grande *stress*, resultando em envelhecimento precoce e subsequentes falhas prematuras. Notadamente, a falha mais comum é dada pelo rompimento do dielétrico entre espiras adjacentes (BONNETT, 1997).

Assim, a proposta aqui apresentada é a aplicação de uma rede MLP para realizar a detecção prematura de curto-circuito entre espiras de motor acionado por conversor de freqüência com PWM senoidal.

C.1 Descrição Geral do Sistema de Detecção de Falha

A detecção de falhas de curto-circuito entre espiras é realizada por uma rede MLP implementada em um processador digital de sinais (DSP), cuja função principal é gerar os pulsos de acionamento dos transistores de potência de um conversor de freqüência. Tal conversor aciona um motor de indução trifásico em freqüências variáveis.

A corrente de uma das fases do motor é aquisitada por um conversor A/D do próprio DSP e, em seguida, utiliza-se a transformada de Fourier para extrair componentes harmônicas específicas do espectro de freqüência da corrente para formar cada vetor de entrada da rede MLP.

O detector neural de falha entre espiras estatóricas do motor de indução trifásico sinaliza visualmente quando da ocorrência de falha.

C.2 Metodologia de Projeto

O desenvolvimento do detector de falhas proposto pode ser dividido em basicamente três etapas:

Simulação Computacional: nesta etapa foram realizadas simulações do motor de indução acionado por conversor de freqüência com o objetivo de gerar dados para a formação e seleção de atributos da rede MLP;

Elaboração de Bancada de Testes: nesta etapa houve o desenvolvimento do conversor de freqüência, a preparação de um motor para simulação de falhas de curto-circuito entre espiras e o acoplamento entre motor e carga;

Ensaios e Validação: nesta etapa são realizados os ensaios para geração do conjunto de dados de treinamento da rede MLP, o próprio treinamento do detector neural e, subseqüentemente, a validação do detector;

C.3 Implementação do Projeto

As três etapas de projeto elencadas na seção C.2 são apresentadas aqui com maior detalhamento.

C.3.1 Simulaçao Computacional

Preliminarmente à confecção da bancada de testes, fez-se mister realizar simulações computacionais da associação conversor-motor, sob diversas condições de carga, com o objetivo de agilizar o processo de avaliação da estratégia adotada para a formação do conjunto de dados de treinamento e teste do detector neural. Como conseqüência imediata, elaborou-se um banco de dados contendo vetores representativos das correntes instantâneas do motor para diversas condições de carga e sob as condições de falha e normalidade do bobinamento estatórico.

As simulações computacionais foram realizadas a partir de uma modelagem[1] em C++ Builder desenvolvida originalmente para a simulação de motores hexafásicos e adaptada para este projeto. A modulação por largura de pulso adotada para o conversor é do tipo simétrica e a freqüência de chaveamento é de 5kHz. As Figuras C.1, C.2 e C.3 ilustram resultados gráficos típicos obtidos para conjungado e correntes do motor sob condição normal e sob falha.

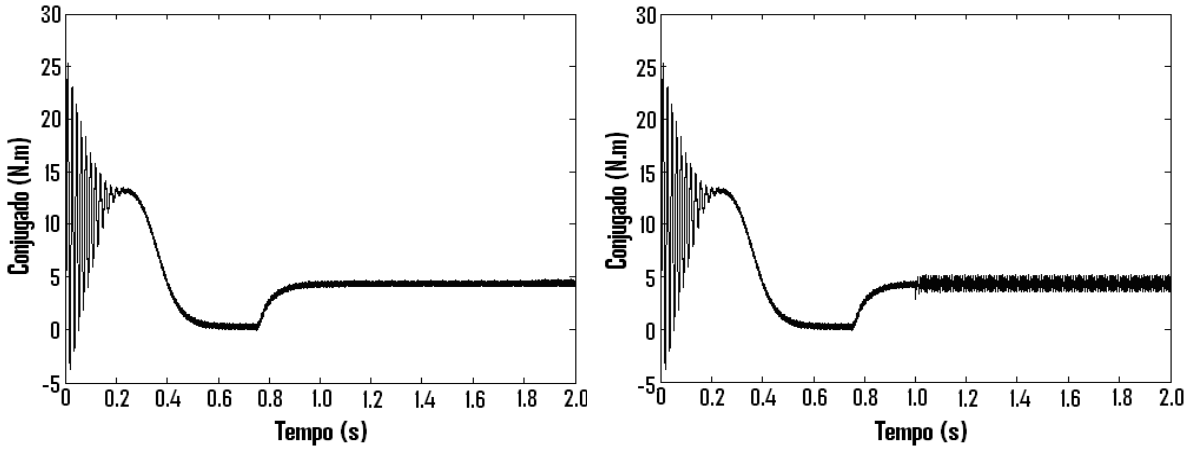

Figura C.1: Conjugado desenvolvido por motor com carga nominal em condições normais e sob falha.

C.3.1.1 Formação dos Conjuntos de Treinamento

Conforme demonstrado por Joksimović & Penman (2000), nenhuma componente nova no espectro de freqüência da corrente do motor aparece como conseqüência da falha no bobinamento estatórico. Na realidade observa-se a ocorrência do crescimento de algumas componentes já existentes. Dessa forma, a proposta inicial é calcular a transformada de

[1] Aplicativo gentilmente disponibilizado pelo Prof. Clayton Ricarte do CEFET-Ce.

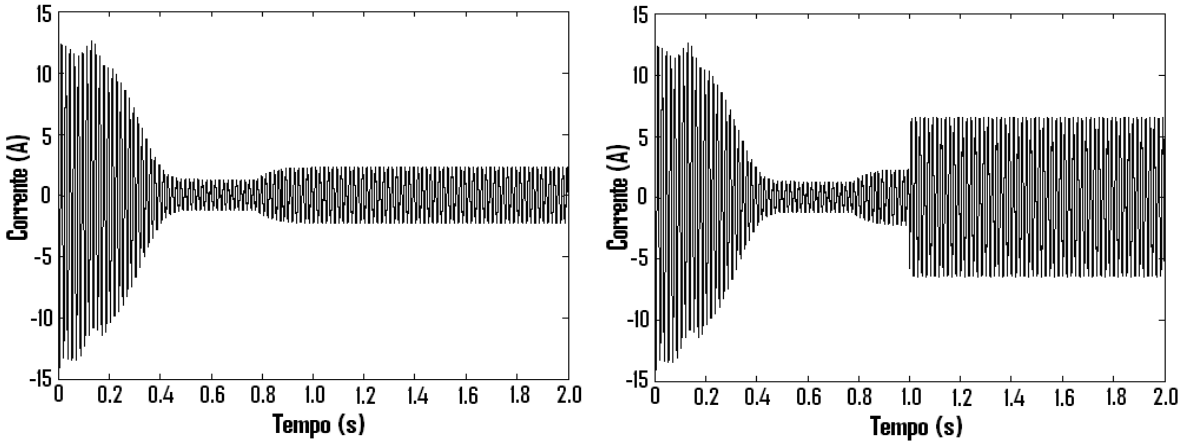

Figura C.2: Corrente na fase-A sob condição normal e sob falha na fase-B.

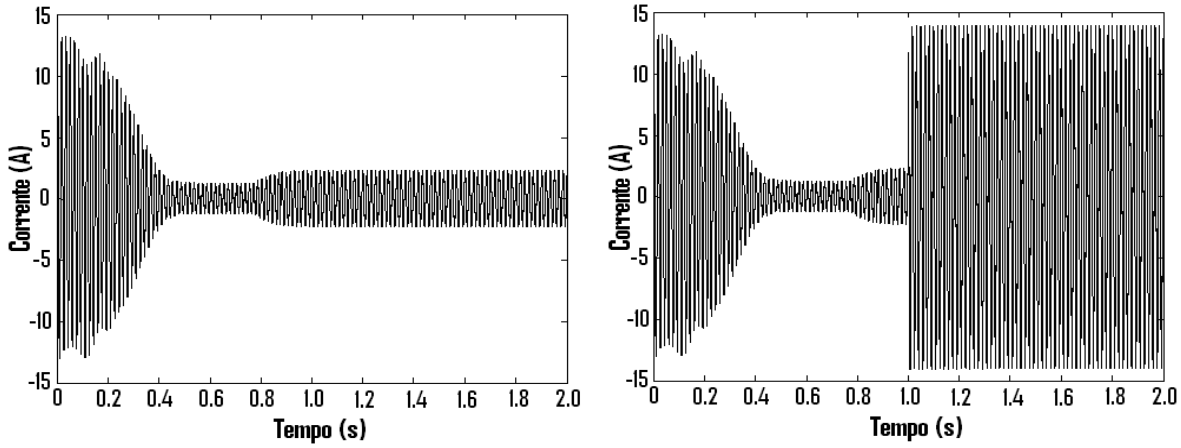

Figura C.3: Corrente na fase-B sob condição normal e sob falha na fase-B.

Fourier de sinais de corrente em diversas condições de carga e falha, e extrair os valores as componentes de freqüência f, $2f$, $3f$, $4f$ e $5f$ para compor os vetores de treinamento da rede MLP. A freqüência f é a componente fundamental do espectro de freqüência da corrente.

Considerando a possibilidade de implementação do detector de falhas numa plataforma experimental baseada na tecnologia de DSP, buscou-se realizar a formação do conjunto de dados, mesmo baseado em simulações computacionais, o mais realista quanto possível. Levou-se em consideração aspectos relacionados com as limitações impostas pelo *hardware* disponível. Assim, optou-se por utilizar uma freqüência de amostragem de 1 kHz e o tamanho de cada vetor de corrente com 2000 pontos. Com isso, 2000 posições de memória RAM de 16 bits devem ser alocadas para armazenar o vetor de corrente; a freqüência do conversor deve ser limitada a valores inferiores a 100 Hz, pois, pelo teorema da

amostragem de Nyquist[2], as componentes de freqüência observadas no sinal não devem ultrapassar a freqüência de Nyquist, no caso 500 Hz; e a discretização do espectro de freqüência é de 0,5 Hz.

É importante observar que com a discretização do espectro de freqüência em 0,5 Hz, não é possível o cálculo das componentes fundamental e algumas de suas múltiplas quando a freqüência fundamental da corrente comandada pelo conversor de freqüência não é múltipla de 0,5 Hz. Assim, ao invés de considerar simplesmente as freqüências f, $2f$, $3f$, $4f$ e $5f$, considera-se intervalos de freqüências centrados nos valores das componentes almejadas.

Foram formados dois conjuntos de dados. O primeiro, denominado Cd_1, é constituído por 38 vetores de 25 atributos. Cada vetor representa uma condição operativa que leva em consideração a freqüência fundamental comandada pelo conversor de freqüência, o percentual de carregamento do motor e a condição do bobinamento estatórico, normal ou curto-circuito de 5% de espiras de uma fase com limitação de corrente de curto-circuito ao valor nominal de corrente do motor. Os vetores relativos à condição normal (19 padrões) são rotulados com $0,98$, enquanto os demais (19 padrões) representam operação com falha no bobinamento e são rotulados com $-0,98$. Os atributos de 1 a 5 estão relacionados com o intervalo de 5 freqüências centrado em f. Os atributos de 6 a 10 estão relacionados com o intervalo de 5 freqüências centrado em $2f$, e assim por diante. Para cada freqüência comandada pelo conversor (30Hz, 40Hz, 43,3Hz, 48,7Hz, 50Hz, 55Hz e 60Hz) foram aplicados níveis de carga diferentes (0%, 25%, 50%, 75% e 100%). Entretanto, as condições operativas simultâneas de bobinamento normal e motor sem carga para cada freqüência citada não são incluídas no conjunto de dados. Elas são utilizadas como referência para a geração dos atributos. Considerando que cada vetor está relacionado a uma freqüência comandada (f_{coman}) específica, cada um de seus atributos representa a variação relativa de uma determinada componente de freqüência tomando como referência o valor desta mesma componente de freqüência calculada para a condição normal a vazio sob a mesma freqüência comandada f_{coman}. Os espectros de freqüência foram calculados a partir de correntes amostradas numa fase que não apresenta falha.

O outro conjunto de dados, denominado Cd_2, é constituído por 76 vetores com 5 atributos. Para cada condição operativa citada no parágrafo anterior foram formados dois vetores: um relativo à corrente de fase normal e o outro relativo à corrente de fase com

[2] Teorema da Amostragem de Nyquist: considerando $x_c(t)$ um sinal limitado em banda com sua versão no domínio da freqüência dado por $X_c(j\Omega) = 0 \quad para \quad |j\Omega| \geq \Omega_N$, então $x_c(t)$ é unicamente determinado por suas amostras $x[n] = x_c(nT)$, $n = 0, \pm 1, \pm 2, \ldots$ se $\Omega_s = \frac{2\pi}{T} \geq 2\Omega_N$. A freqüência Ω_N é comumente denominada como a freqüência de Nyquist (OPPENHEIM et al., 1999).

curto-circuito entre espiras (motor conectado em estrela). Cada atributo representa o valor médio dos coeficientes das componentes de freqüência de um intervalo de 3 componentes centrados em f, $2f$, $3f$, $4f$ e $5f$.

C.3.1.2 Seleção de Características

De posse dos conjuntos de dados formados na subseção anterior, pretende-se estimar o número de atributos mínimo necessário e suficiente para compor os vetores de entrada do detector neural de falhas. Para isto, redes MLP são treinadas com os algoritmos de retropropagação dos erros e *Levenberg-Marquardt*, e podadas com o auxílio do algoritmo CAPE, funcionando como método de seleção de características.

O conjunto de dados Cd_1 foi utilizado para o treinamento inicial de uma rede MLP com 25 unidades de entrada, 5 neurônios ocultos e 1 neurônio de saída. Os dados foram pré-processados pela remoção da média e divisão pelo desvio padrão de cada atributo. O treinamento foi realizado com a aplicação do algoritmo de retropropagação iterativo com uma taxa de aprendizagem de $0,01$ e fator de momento de $0,05$ até atingir 100% de acerto. Em seguida, a aplicação do CAPE encontrou 4 atributos redundantes, como pode ser visto na Tabela C.1.

Tabela C.1: Resultados de seleção progressiva de características em conjunto de dados com 25 atributos para classificação de falhas estatóricas.

	Q	N_c	CR_{train}	ε_{train}	N_i	ATR_{elim}
Arquitetura 1	5	136	100	0,0349	25	-
CAPE	3	45	100	0,0497	21	1, 16, 21, 24
Arquitetura 2	5	116	100	0,0031	21	-
CAPE	4	57	100	0,0520	20	17
Arquitetura 3	5	111	100	0,0008	20	-
CAPE	4	39	100	0,0622	17	22, 23, 25
Arquitetura 4	5	96	100	0,0007	17	-
CAPE	5	49	100	0,0137	16	4
Arquitetura 5	5	91	100	0,0235	16	-
CAPE	5	50	100	0,0491	15	3
Arquitetura 6	5	86	100	0,0001	15	-
CAPE	4	42	100	0,0502	13	2, 10
Arquitetura 7	5	76	100	0,0001	13	-

Diante deste resultado, formou-se um novo conjunto de dados, agora com a exclusão dos atributos 1, 16, 21 e 24, perfazendo apenas $N_i = 21$ atributos e treinou-se outra rede MLP e aplicou-se a poda. Este procedimento foi repetido progressivamente até que não

houvesse mais atributos a serem eliminados. Ao final, restaram apenas 13 características. Todas as componentes relacionadas com $5f$ foram excluídas e apenas uma componente relacionada com f restou. As componentes relacionadas com $2f$, $3f$ e $4f$ se mostram importantes para a classificação. Para comprovar isto, uma última rede (Arquitetura 7) foi treinada com 13 unidades de entrada, 5 neurônios ocultos e 1 neurônio de saída, atingindo 100% de acerto na classificação.

O conjunto de dados Cd_2 foi utilizado para o treinamento inicial de uma rede MLP com 5 unidades de entrada, 5 neurônios ocultos e 1 neurônio de saída. Os dados foram pré-processados pela remoção da média e divisão pelo desvio padrão de cada atributo. Tentativas de treinamento com o algoritmo de retropropagação iterativo não resultou em 100% de acerto na classificação. Assim, optou-se pela aplicação do algoritmo *Levenberg-Marquardt*, e com isso atingiu-se $CR_{train} = 100\%$ de acerto e $\varepsilon_{train} = 0,001$ de erro.

A aplicação do CAPE encontrou o primeiro atributo como redundante (Tabela C.2). Então, treinou-se diversas redes MLP, agora com apenas 4 atributos, até atingir $CR_{train} = 100\%$ de acerto e subsequentemente aplicou-se O CAPE. Em nenhuma das simulações foi possível extrair qualquer atributo ou reduzir o número de neurônios ocultos. A topologia resultante da aplicação do CAPE à Arquitetura 2 (Tabela C.2) foi melhor o resultado dentre todas as simulações.

Tabela C.2: Resultados de seleção progressiva de características em conjunto de dados (resultante de simulação computacional) com 5 atributos para classificação de falhas estatóricas.

	Q	N_c	CR_{train}	ε_{train}	N_i	ATR_{elim}
Arquitetura 1	5	36	100	0,001	5	-
CAPE	5	30	100	0,0028	4	1
Arquitetura 2	5	31	100	0,0008	4	-
CAPE	5	26	100	0,0013	4	-

A Tabela C.5 resume resultados alcançados com a seleção de atributos, enfocando aspectos relativos à implementação para os dois conjuntos de dados. Seguem algumas observações:

- Apenas 12 componentes de freqüência são necessárias no conjunto de dados Cd_2, enquanto no conjunto Cd_1 são necessárias 13;
- A menor topologia alcançada na Tabela C.1 foi obtida com a aplicação do CAPE à Arquitetura 6. Nesta, apesar de restarem apenas 42 pesos, as matrizes de pesos W1

(pesos da camada oculta) e W2 (pesos da camada de saída) representativas desta topologia requerem a armazenagem de 61 valores, incluindo os zeros. Já no caso da Tabela C.2, a menor topologia, além de apresentar apenas 26 pesos, requer a armazenagem de apenas 31 valores;

- O tempo de processamento da topologia ancançada com o conjunto de dados Cd_2 é menor. Considera-se aqui o número total de operações matemáticas necessárias para a execução da classificação de um único padrão de entrada. Não é feita qualquer distinção entre operações de adição, multiplicação ou cálculo de valores da tangente hiperbólica.

Tabela C.3: Tabela comparativa dos resultados finais da seleção de características nos conjuntos de dados Cd_1 e Cd_2.

Descrição	Cd_1	Cd_2
Número de Atributos	13	12
Dimensão de W1	4 x 14	5 x 5
Dimensão de W2	1 x 5	1 x 6
Posições de Memória	61	31
Operações da MLP	39	24

A avaliação dos resultados do estudo realizado sugere a utilização de apenas 12 componentes de freqüência do espectro de freqüência de uma das correntes do motor de indução trifásico como base para formação dos atributos dos vetores de entrada do classificador de falhas neural proposto. Cada atributo é composto pela média das componentes de freqüência calculada em cada intervalo de 3 componentes de freqüência centrado em $2f$, $3f$, $4f$ e $5f$.

C.3.2 A Bancada de Testes

A bancada de testes é composta por um conversor de freqüência executado com a tecnologia de DSP, um conjunto motor-carga e um sistema de aquisição de dados. O Motor é especialmente preparado para permitir a simulação de falha entre espiras e o sistema de aquisição de dados é implementado no próprio conversor.

C.3.2.1 Conversor de Freqüência

A Figura C.4 mostra os principais componentes do conversor de freqüência implementado. Na parte superior da figura pode-se ver a placa de potência, onde se encontram o

conversor CA-CC não controlado com filtro capacitivo, a ponte de IGBTs (transistores) que compõe o inversor, e os sensores de corrente do tipo efeito Hall. Na parte central encontra-se a placa de interface de potência, onde se observa os *drivers* para os transistores do inversor e as fontes chaveadas auxiliares dos *drivers*. Na parte inferior, à esquerda, encontra-se uma placa de desenvolvimento com o processador digital de sinais (DSP) TMS320F2812 da Texas Instruments. O processador realiza os cálculos para a geração dos pulsos para o inversor, além de calcular o espectro de freqüência de uma das correntes estatóricas e implementar a detecção de falha com uma rede MLP. Na parte inferior da figura ainda podem ser vistas a placa de interface digital (ao centro) entre o DSP e os *drivers* dos IGBTs, e a fonte linear para polarização dos sensores de corrente (à direita). Embora não constem na figura, dois outros componentes ainda fazem parte do conversor. O primeiro é uma placa com os circuitos condicionadores dos sinais analógicos das correntes das três fases de alimentação do motor e da tensão do barramento CC. O segundo é uma interface homem-máquina (IHM) que disponibiliza aos usuários informações sobre o funcionamento do conversor e permite a alteração de alguns parâmetros de programação relativos à operação do mesmo.

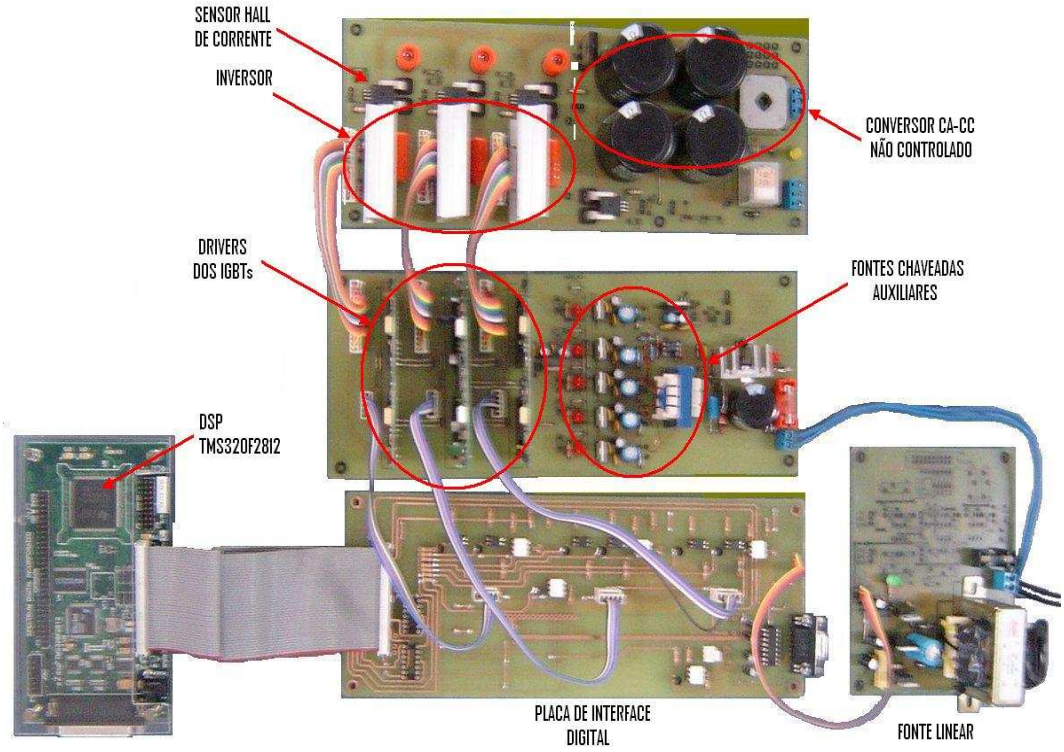

Figura C.4: Conversor de freqüência, CPU e circuitos auxiliares.

O conversor foi projetado para ser alimentado por fonte senoidal monofásica com tensão eficaz de 220V. A freqüência das tensões trifásicas de saída pode variar de 0 a

60Hz, com a manutenção da relação V/f constante. Dessa forma, o motor a ser acionado deve ser ligado na configuração delta. Os pulsos de comando das chaves de potência são gerados numa freqüência de 10kHz através da técnica de combinação de vetores espaciais (*Space Vectors*).

C.3.2.2 O Motor

A máquina utilizada neste projeto é um motor de indução trifásico de $0,75$kW ($1,0$CV), marca WEG, com velocidade nominal de 1720rpm, rendimento de $79,5\%$, fator de potência 0.82, tensões 220/380V e correntes $3,02/1,75$A.

Originalmente cada fase do motor é composta por 2 grupos de 3 bobinas com 58 espiras, perfazendo um total de 348 espiras (Figura C.5). Apenas 2 terminais por fase são disponibilizados para a ligação do motor.

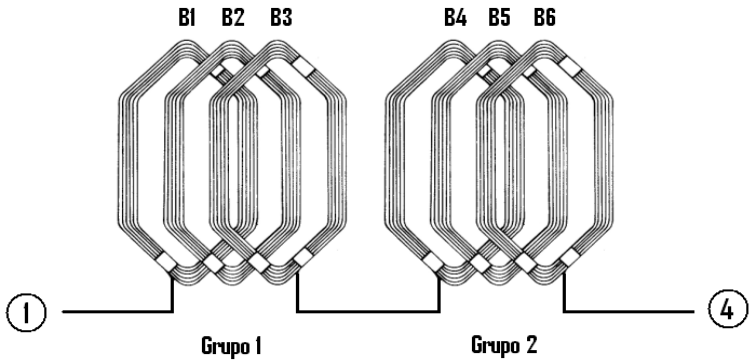

Figura C.5: Bobinamento estatórico da fase A do MIT.

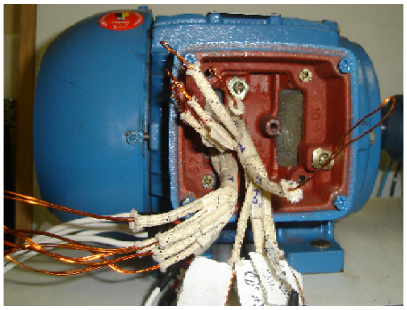

Figura C.6: Motor de indução trifásico rebobinado.

Após a execução de serviço de rebobinamento em empresa especializada, derivações da bobina B1 (grupo 1) de cada fase do motor estão disponíveis para a realização de simulações de diversas condições de curto-circuito entre espiras no motor. O rebobinamento foi realizado de tal forma que se mantivesse as características do bobinamento o

mais próximo quanto possível do original. A Figura C.6 mostra o motor rebobinado, com destaque para a caixa de ligação dos terminais, onde se pode ver as derivações de B1.

As Figuras C.7, C.8 e C.9 mostram três possíveis condições de simulação de falha entre espiras. Na Figura C.7, a falha é simulada apenas na espira 2, enquanto nas outras duas figuras se tem, respectivamente, 3 e 7 espiras sob simulação de falha. É importante notar que a falha é simulada com a inserção de resistor limitador em série com as espiras sob simulação de falha. Caso contrário, o bobinamento pode ser destruído devido as elevadas correntes de curto-circuito. Esta condição limitada de teste pode perfeitamente representar um estágio intermediário no desenvolvimento da falha, tendo em vista que geralmente o curto-circuito entre espiras é iniciado com uma falha de alta impedância (NATARANJAN, 1989).

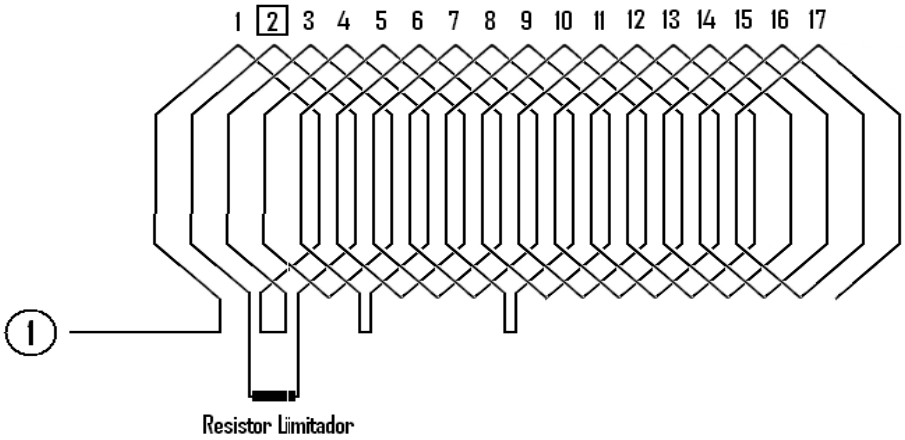

Figura C.7: Bobinamento com a espira 2 em curto-circuito.

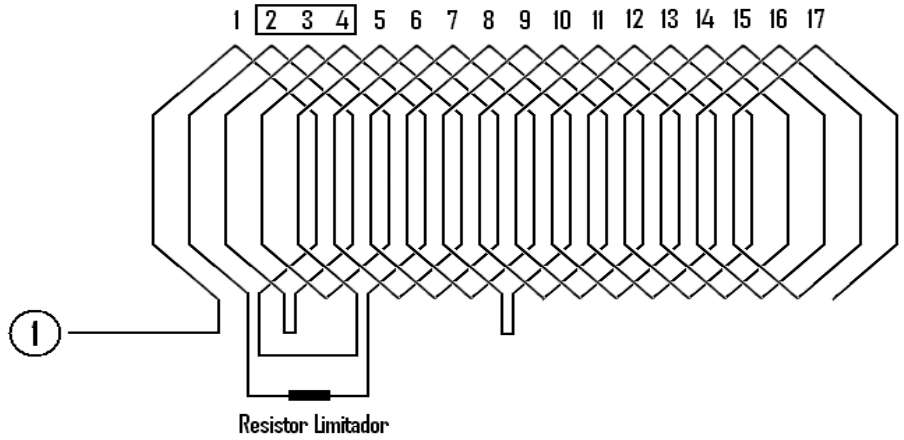

Figura C.8: Bobinamento com as espiras 2, 3 e 4 em curto-circuito.

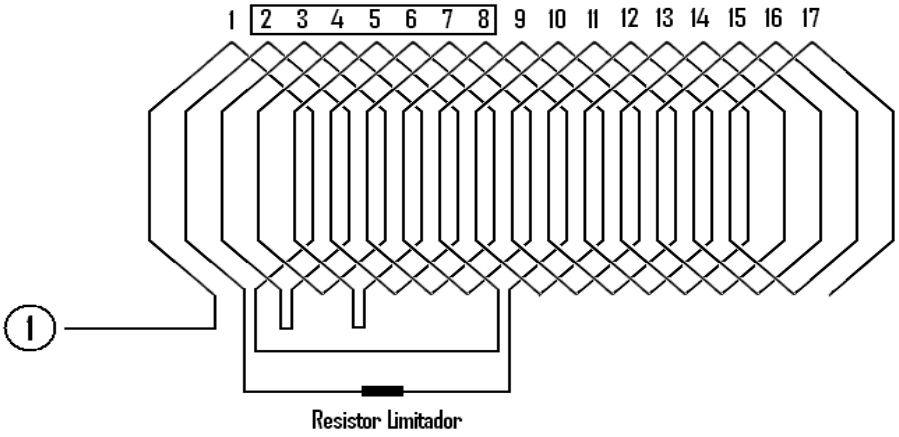

Figura C.9: Bobinamento com as espiras 2, 3, 4, 5, 6, 7 e 8 em curto-circuito.

C.3.2.3 Conjunto Motor-Carga

O detector de curto-circuito entre espiras deve ser capaz de acusar a falha sob diversas condições de carga do motor. Dessa forma, optou-se por acoplar um eletro-dinamômetro ao eixo do motor sob ensaio.

O eletro-dinamômetro empregado nada mais é do que outro motor de indução trifásico com as mesmas especificações do motor sob ensaio, mas sendo alimentado por um conversor CA-CC totalmente controlado com diodo roda-livre. O aumento do ângulo de condução do conversor CA-CC produz aumento de carga para o motor sob ensaio.

A Figura C.10 mostra o arranjo mecânico motor-carga.

Figura C.10: Motor acoplado a um eletro-dinamômetro.

C.3.3 Ensaios e Resultados

Com a estratégia definida para a formação do conjunto de dados de treinamento, conforme a subseção C.3.1.2, e com a bancada de testes concluída, realizaram-se diversos ensaios para a aquisição dos sinais de corrente do motor sob as condições de falha e, por outro lado, de normalidade do bobinamento estatórico.

Foram realizados ensaios sob as seguintes condições:

- Ligação dos terminais do motor: delta;
- Freqüências comandadas: $30,83$Hz, $39,72$Hz, $42,7$Hz, 50Hz e $59,72$Hz;
- Condições de carga: 0% e 50%;
- Freqüência de amostragem: 1kHz;
- Tamanho do vetor de corrente: 1000 amostras;
- Extensão do curto-circuito: 6% de espiras;
- Limitação de corrente de curto-circuito ao valor nominal de corrente do motor.

A Figura C.11 ilustra o conteúdo da memória RAM de 16 bits (variável do tipo *unsigned int*) do DSP relativo à forma de onda de corrente aquisitada na fase-B do conversor de freqüência com o motor sob condição de falha de curto-circuito entre espiras e carga de 50%. A freqüência e a tensão eficaz comandadas pelo conversor são $59,72$Hz e 225V, respectivamente. A corrente eficaz da fase-B do conversor é de $2,5$A e a corrente de curto-circuito é de $1,7$A. Diante da constatação de ruído no sinal aquisitado optou-se pela implementação de um filtro digital adicional. Calcula-se a média ($\bar{y}$) e o desvio padrão (σ_y) de cada sinal e substitui-se qualquer amostra deste mesmo sinal que ultrapasse o intervalo ($\bar{y} - 1,3.\sigma_y$, $\bar{y} + 1,3.\sigma_y$) pela média de suas duas amostras adjacentes. Pode-se ver na Figura C.11 que há uma remoção satisfatória do ruído.

Para cada freqüência comandada, sob cada condição de carga e de integridade do bobinamento, foram obtidos dois sinais de corrente. Um relativo a uma fase diretamente ligada à falha e outro não. Assim, o conjunto de dados é composto por 20 padrões relativos à condição normal (rotulado com $0,98$) e 20 padrões relativos à condição de falha (rotulado com $-0,98$), perfazendo um total de 40 padrões.

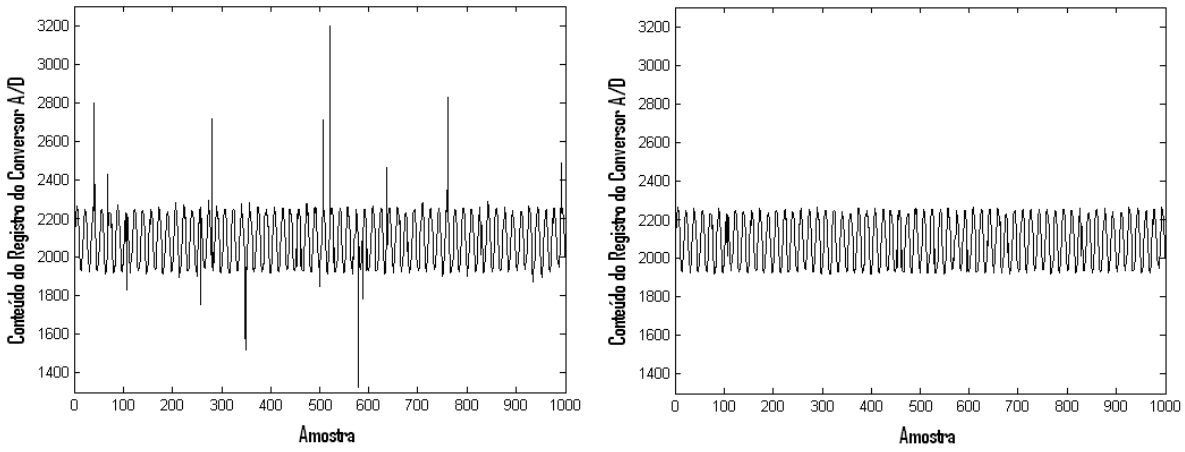

Figura C.11: Corrente na fase-B sob condição de falha com 50% de carga antes (esquerda) e após (direita) a filtragem digital.

Conforme sugerido pelos resultados da simulação computacional, cada padrão é composto por 5 atributos. Mas aqui, os atributos são os coeficientes de correlação entre as 5 primeiras componentes de freqüência múltiplas da freqüência comandada pelo conversor (f, $2f$, $3f$, $4f$ e $5f$) e o sinal de corrente em questão. É importante salientar que em C.3.1.1 as componentes de freqüência são computadas pelo algoritmo *Fast Fourier Transform* (FFT), o que fornece uma discretização do espectro de freqüência de $0,5$Hz, justificando assim a formação de cada atributo pelo valor médio dos coeficientes das componentes de freqüência em um intervalo de 3 componentes centrados em f, $2f$, $3f$, $4f$ e $5f$. Aqui, cada componente é computada pelo algoritmo mostrado em C.4.

De posse do conjunto de dados, aplicou-se procedimento semelhante ao utilizado em C.3.1.2 para estimar o número de atributos mínimo necessário e suficiente para a perfeita classificação dos padrões. Inicialmente, redes MLP ($P = 5$, $Q = 5$, $M = 1$) são treinadas com dados pré-processados (remoção da média e divisão pelo desvio padrão de cada atributo) e com a utilização do algoritmo *Levenberg-Marquardt*. Em seguida, as redes são podadas com o algoritmo CAPE. Os resultados podem ser vistos na Tabela C.5.

Seguindo a tendência encontrada nas simulações computacionais, o primeiro atributo, ou seja a freqüência fundamental, foi considerado redundante. A aplicação sucessiva de treinamento e poda, além de selecionar características, também resultou numa topologia menor ($P = 4$, $Q = 3$, $M = 1$). Esta configuração foi alcançada com a poda aplicada à Arquitetura 3 (Tabela C.5) e confirmada com o treinamento da Arquitetura 4.

A Figura C.12 mostra as projeções bidimensionais dos atributos aplicados à rede MLP resultante da poda realizada sobre a Arquitetura 3. Aparentemente as classes não são

Tabela C.4: Procedimento de cálculo dos coeficientes de correlação das componentes de freqüência de interesse.

1. Definições preliminares
   y // `sinal de corrente`
   $f = f_{comandada}$; // `freqüência comandada`
   $N = 1000$; // `número de amostras`
   $ta = 0,001$; // `período de amostragem`
   $m = 2 * pi * ta$; // `constante auxiliar`
2. Definição das componentes de freqüência
   $cfreq(1) = m * f$; // `primeira harmônica`
   $cfreq(2) = m * 2 * f$; // `segunda harmônica`
   $cfreq(3) = m * 3 * f$; // `terceira harmônica`
   $cfreq(4) = m * 4 * f$; // `quarta harmônica`
   $cfreq(5) = m * 5 * f$; // `quinta harmônica`
3. Cálculo dos coeficientes de correlação

```
for k = 1 : 5,
    Ssen = 0;
    Scos = 0;
    for i = 1 : N,
        Scos = Scos+y(i)*cos(cfreq(k)*i);
        Ssen = Ssen+y(i)*sin(cfreq(k)*i);
    end
atributo(k) = (sqrt(Scos ^ 2+Ssen ^ 2))/N;
end
```

Tabela C.5: Resultados de seleção progressiva de características em conjunto de dados (resultante de ensaios) com 5 atributos para classificação de falhas estatóricas.

	Q	N_c	CR_{train}	ε_{train}	N_i	ATR_{elim}
Arquitetura 1	5	36	100	0,0218	5	-
CAPE	4	21	100	0,0455	5	-
Arquitetura 2	4	29	100	0,0298	5	-
CAPE	4	21	100	0,0569	4	1
Arquitetura 3	4	25	100	0,0453	4	-
CAPE	3	19	100	0,0442	4	-
Arquitetura 4	3	19	100	0,0583	4	-
CAPE	3	17	100	0,0796	4	-

separáveis. Entretanto, como pode ser visto na Figura C.13, os neurônios da camada oculta projetam os dados em um espaço tridimensional, no qual há a perfeita separação das classes com os dados de treinamento, justificando assim a utilização da rede MLP como ferramenta para modelagem não-linear.

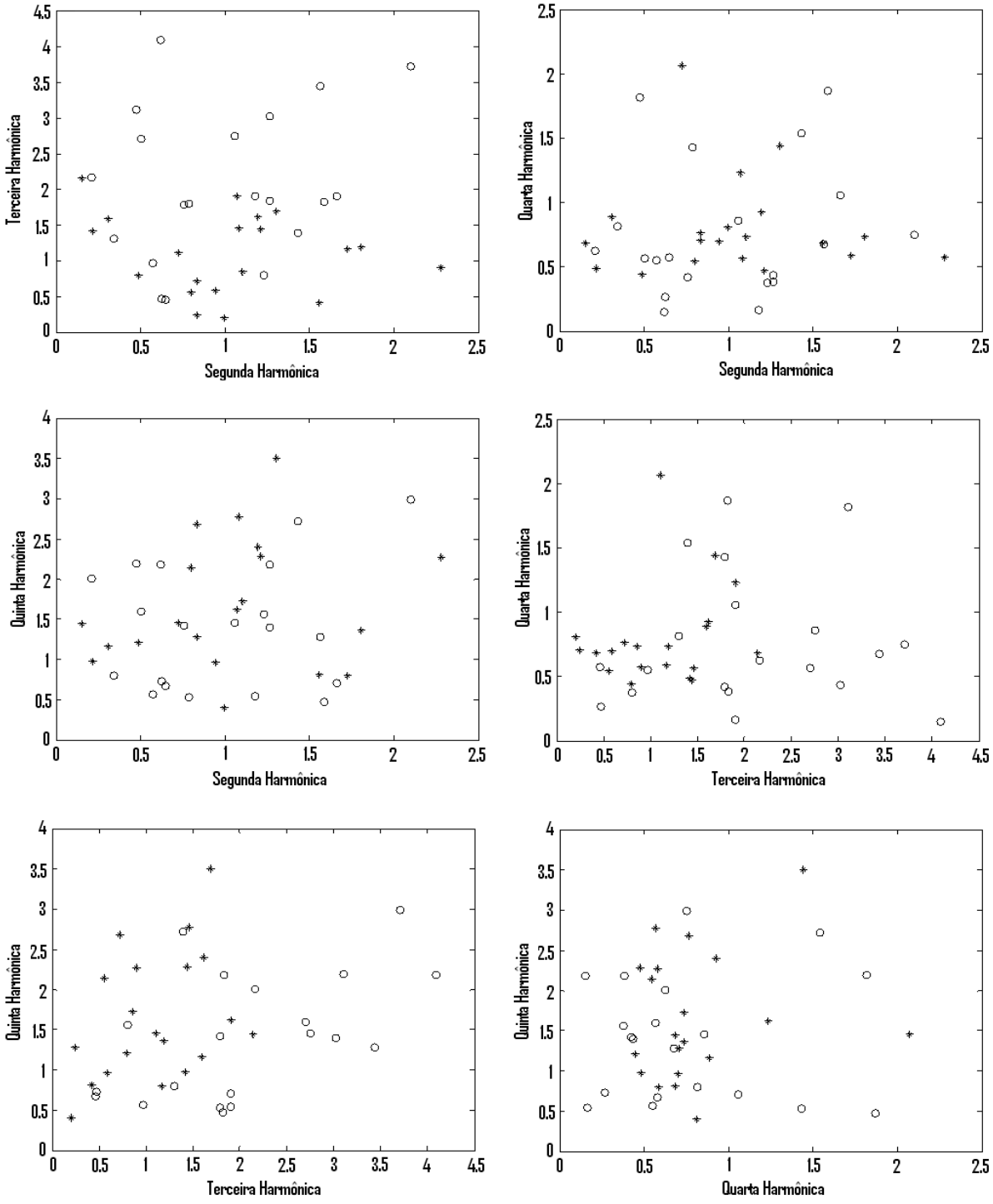

Figura C.12: Projeções bidimensionais dos atributos aplicados ao detector neural de falhas de curto-circuito entre espiras. (*) bobinamento normal (o) bobinamento com falha.

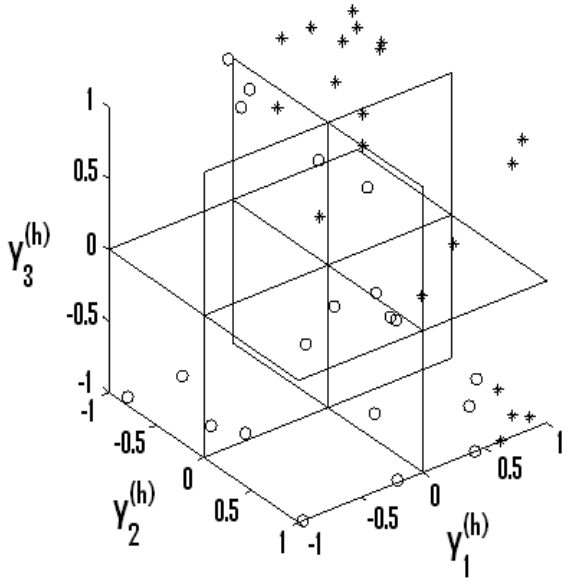

Figura C.13: Organização dos dados no espaço tridimensional projetado pelos neurônios ocultos. (*) bobinamento normal (o) bobinamento com falha.

C.4 Conclusões

O detector neural de falhas de curto-circuito entre espiras de motor de indução trifásico acionado por conversor de freqüência foi implementado com sucesso.

Foram realizadas previamente simulações computacionais com o objetivo de estudar a influência dos atributos escolhidos para formar o conjunto de dados sobre a capacidade de classificação da rede MLP utilizada. Em seguida, realizaram-se ensaios em motor especialmente preparado para simular as referidas falhas para a obtenção dos sinais de corrente estatórica sob condições normais e sob condições de falhas. Estas amostras foram utilizadas para compor os atributos do conjunto de dados.

Finalmente, o detector neural foi projetado e implementado com sucesso a partir dos dados obtidos com os ensaios. Os resultados alcançados com a experimentação prática confirmaram àqueles obtidos com as simulações.

APÊNDICE D – Custo Computacional CAPE Versus ***OBS***

O custo computacional é uma característica importante a ser observada na análise técnica de viabilidade da aplicação de algorítmos. Muitas vezes a efetividade de um algoritmo na solução de problemas vem acompanhada pela execução de tarefas complexas e/ou uso excessivo de recursos de memória, o que pode inviabilizar sua aplicação em tempo real ou em sistemas de baixo custo.

Nesta tese o CAPE é apresentado como um método alternativo de fácil implementação para a poda de MLPs. A sua efetividade já foi comprovada através de simulações computacionais no capítulo 4, onde é realizado um estudo comparativo de seu desempenho (detecção e eliminação de redundância de conexões sinápticas) em relação a um algoritmo bem estabelecido, o OBS. Aqui, é apresentado outro estudo comparativo entre os dois algoritmos acima citados. Mas desta vez, o foco é o número de operações matemáticas executadas. Para isto, foi realizado o levantamento de todas as operações matemáticas necessárias para a obtenção das sensibilidades (S_i) dos pesos sinápticos no algoritmo OBS, assim como dos índices de correlação cruzada (C_{oh} e C_{hi}) no algoritmo CAPE.

As Tabelas D.1 e D.2 mostram o número de operações matemáticas para ambos os algoritmos em função do número de amostras no conjunto de dados (N), do número de unidades de entrada (P), do número de neurônios ocultos (Q) e do número de neurônios de saída (O). Percebe-se que predomina uma relação quadrática entre as variáveis acima citadas e o número de operações matemáticas no algoritmo OBS, enquanto no CAPE, a predominância é linear.

Por questão de simplicidade e facilidade de visualização pode-se relacionar o número total de operações N_{op}, indistintamente se são multiplicações, somas e etc., com o número de conexões sinápticas ($N_c = (P+1)Q + (Q+1)O$). A expressão geral de $N_{op}(N_c)$ para

Tabela D.1: Operações Matemáticas no Método CAPE.

Operação	Número de Operações
Soma	$N.[3PQ + 3Q.O - P + 2Q + 3O - 2] - PQ - QO - O - 1$
Subtração	$N.[Q + 2O]$
Multiplicação	$N.[4PQ + 4QO + 5Q + 5O]$
Divisão	-
Tangente hiperbólica	$N.[Q + O]$

Tabela D.2: Operações Matemáticas no Método OBS.

Operação	Número de Operações
Soma	$N.[3P^2Q^2 + 3P^2Q^2.O + 3Q^2O^2 + 2QO + Q + O]$
Subtração	$N.[2PQ^2O + P^2Q^2 + Q^2O^2 + PQ + 4QO + Q + 2O]$
Multiplicação	$N.[8PQ^2O + 4P^2Q^2 + 4Q^2O^2 + 8PQ + 11QO + 3Q + 3O]$
Divisão	$N.[P^2Q^2 + 2PQ^2O + Q^2O^2 + PQ + QO]$
Tangente hiperbólica	$N.[Q + O]$

ambos os algoritmos é dada pela Equação

$$N_{op}(N_c) = a.N_c^2 + b.N_c, \tag{D.1}$$

onde os valores de a e b podem ser estimados para cada problema particularmente.

A Figura D.1 mostra a dependência quadrática em relação a N_c do custo computacional ($N_{op\_obs}(N_c)$) do algoritmo OBS e a dependência linear do custo operacional ($N_{op\_cape}(N_c)$) do algoritmo CAPE para a poda de uma rede MLP com duas unidades de entrada ($P = 2$), uma única saída ($O = 1$) e o número de neurônios ocultos (Q) variando de um a dez. O número de amostras no conjunto de treinamento (N) utilizado para calcular as sensibilidades no OBS e os índices de correlação cruzada no CAPE é 140. É importante observar que o custo computacional da aplicação do algoritmo OBS cresce rapidamente com o crescimento do número de neurônios na camada escondida, o que não acontece com o algoritmo CAPE. Isto pode constituir um fator de decisão importante a favor do algoritmo CAPE na poda de MLPs que se prestam a modelar problemas complexos (muitos neurônios ocultos).

A Tabela D.3 mostra os parâmetros a e b referentes à estimativa da dependência do custo computacional em função do número de conexões sinápticas (Equação D.1) para a aplicação dos algoritmos de poda CAPE e OBS nas diversas simulações apresentadas nesta tese.

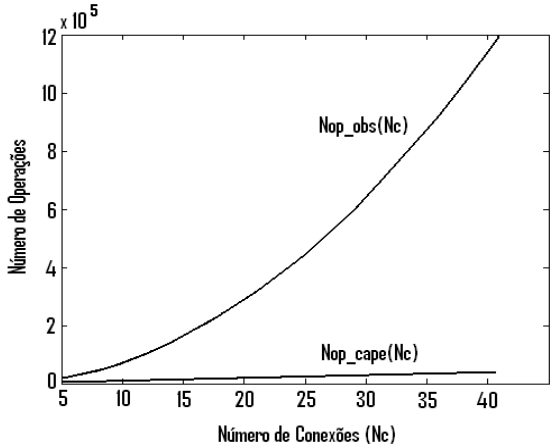

Figura D.1: Custo computacional CAPE x OBS para a poda de MLPs com 2 unidades de entrada, 1 neurônio de saída e número variável de neurônios ocultos.

Tabela D.3: Estimativas dos parâmetros a e b da Equação D.1.

				CAPE	OBS
N	P	O	Q	$a = 0$, b	a , $b = 0$
140	2	1	1-10	1050	700
105	4	3	1-10	775	800
99	13	3	1-10	700	1200
60	34	6	1-10	425	1200
144	6	3	1-10	1050	1300

APÊNDICE E – Tabelas de Características dos Conjuntos de Dados

Tabela E.1: Conjunto de dados Iris.

Planta (número de amostras)	Atributos
Iris Setosa(50)	1: comprimento da sépala (cm)
Iris Versicolor(50)	2: largura da sépala (cm)
Iris Virgínica(50)	3: comprimento da pétala (cm)
	4: largura da pétala (cm)

Tabela E.2: Conjunto de dados Wine.

Vinho (número de amostras)	Atributos
Produtor 1(59)	1: álcool
Produtor 2(71)	2: ácido málico
Produtor 3(48)	3: cinza
	4: alcalinidade da cinza
	5: magnésio
	6: total de fenóis
	7: flavanóides
	8: fenóis não-flavanóides
	9: proanthocyanins
	10: intensidade da cor
	11: tonalidade
	12: OD280/OD315 de vinhos diluídos
	13: Prolina

Tabela E.3: Conjunto de dados biomédicos relacionados a doenças da coluna vertebral.

Condição da coluna (número de amostras)	Atributos
Normal(100)	1: incidência pélvica
Hérnia de Disco (60)	2: inclinação pélvica
Spondilolistese(150)	3: angulação sacral
	4: raio pélvico
	5: ângulo de lordose lombar
	6: grau de spondilolistese

Tabela E.4: Conjunto de dados Dermatology.

	Atributos	
Doença (número de pacientes)	Clìnicos	Histopatológicos
Psoríase(111)	1: eritema	12: incontinência de melanina
Dermatite seborréica(60)	2: escala	13: eosinófilos no infiltrado
Líquen plano(71)	3: bordas definidas	14: infiltrado PNL
Pitiríase rósea(48)	4: coceira	15: fibrose na derme papilar
Dermatite crônica(48)	5: fenômeno de Koebner	16: exocitose
Pitiríase rubra pilar(20)	6: pápulas poligonais	17: acantose
	7: pápulas foliculares	18: hiperceratose
	8: envolvimento da mucosa oral	19: paraceratose
	9: envolvimento do joelho e do cotovelo	20: dilatação em clava dos cones epiteliais
	10: envolvimento do escalpo	21: alongamento dos cones epiteliais da epiderme
	11: histórico familiar	22: estreitamento da epiderme suprapapilar
	34: idade	23: pústulas espongiformes
		24: microabscesso de Munro
		25: hipergranulose focal
		26: ausência da camada granulosa
		27: vacuolização e destruição da camada basal
		28: espongiose
		29: aspecto “dente de serra” das cristas interpapilares
		30: tampões cárneos foliculares
		31: paraceratose perifolicular
		32: infiltrado inflamatório mononuclear
		33: infiltrado em banda

Referências

AKAIKE, H. Statistical predictor identification. *Annals of the Institute of Statistical Mathematics*, v. 22, n. 2, p. 203–217, 1970.

AKAIKE, H. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, v. 19, n. 6, p. 716–723, 1974.

BARBOUR, A.; THOMSON, W. T. Finite element study of rotor slot designs with respect to current monitoring for detecting static airgap eccentricity in squirrel-cage induction motors. *IEEE Industry Application*, p. 112–119, 1997.

BARRON, A. Universal approximation bounds for superposition of sigmoids functions. *IEEE Transactions on Information Theory*, v. 39, n. 3, p. 930–945, 1993.

BARRON, A. R. Predicted squared error: A criterion of automatic model selection. *Self-Organizing Methods in Modeling*, 1984.

BAUM, E. B.; HAUSSLER, D. What size net gives valid generalization? *Neural Computation*, v. 1, p. 151–160, 1989.

BELL, S.; SUNG, J. Will your motor insulation survive a new adjustable-frequency drive? *IEEE Transactions on Industry Applications*, v. 33, n. 5, p. 1307–1311, 1997.

BELLINI, A. et al. Quantitative evaluation of induction motor broken pars by means of electrical signature analysis. *IEEE Transactions on Industry Applications*, v. 37, n. 5, p. 1248–1255, 2001.

BENGTSSON, T.; CAVANAUGH, J. E. An improved akaike information criterion for state-space model selection. *Computational Statistics & Data Analysis*, v. 50, p. 2635–2654, 2006.

BERRY, M. J. A.; LINOFF, G. *Data Mining Techniques*. [S.l.]: John Willey and Sons, 1997.

BERTHONNAUD, E. et al. Analysis of the sagittal balance of the spine and pelvis using shape and orientation parameters. *Journal of Spinal Disorders & Techniques*, v. 18, n. 1, p. 40–47, 2005.

BHAYA, A.; KASZKUREWICZ, E. Steepest descent with momentum for quadratic functions is a version of the conjugate gradient method. *Neural Networks*, v. 17, n. 1, p. 65–71, 2004.

BISHOP, C. M. Exact calculation of the hessian matrix for the multi-layer perceptron. *Neural Computation*, v. 4, n. 4, p. 494–501, 1992.

BISHOP, C. M. *Neural Networks for Pattern Recognition*. [S.l.]: Oxford University Press, 1995.

BLUM, A. *Neural Networks in C++*. [S.l.]: Wiley, 1992.

BONNETT, A. H. A comparison between insulation systems available for pwm-inverter-fed motors. *IEEE Transactions on Industry Applications*, v. 33, n. 5, p. 1331–1341, 1997.

BONNETT, A. H.; SOUKUP, G. Cause and analysis of stator and rotor failures in three-phase squirrel-cage induction motors. *IEEE Transaction on Industry Applications*, v. 28, n. 4, p. 921–937, 1992.

BOSE, N. K.; GARGA, A. K. Neural network design using Voronoi diagrams. *IEEE Transactions on Neural Networks*, v. 4, n. 5, p. 778–787, 1993.

CARDOSO, A. J. M.; SARAIVA, E. S. Computer-aided detection of airgap eccentricity in operating three-phase induction motors by park's vector approach. *IEEE Transactions on Industry Applications*, v. 29, n. 5, p. 897–901, 1993.

CARUANA, R. Generalization vs. network size. In: *Proceedings of the NIPS Workshop on Generalization*. [S.l.: s.n.], 1993.

CASH, M. A.; HABETLER, T. G.; KLIMAN, G. B. Insulation failure prediction in induction machines using line-neutral voltages. In: *Proceedings of the IEEE Conference on Industry Applications (IAS'97)*. [S.l.: s.n.], 1997. v. 1, p. 208–212.

CHAUVIN, Y. A back-propagation algorithm with optimal use of hidden units. In: TOURETZKY, D. S. (Ed.). *Advances in Neural Network Processing*. [S.l.]: Morgan Kaufmann, 1989. v. 1, p. 519–526.

CHEN, Y.; HU, S.; CHEN, D. Fast pruning strategy for neural network size optimization and its applications. *Journal of Chemical Industry and Engineering*, p. 522–526, 2001.

CORANI, G.; GUARISO, G. An application of pruning in the design of neural networks for real time flood forecasting. *Neural Computing and Applications*, v. 14, n. 1, p. 66–77, 2005.

CRAVEN, P.; WAHBA, G. Smoothing noisy data with spline functions: Estimating the correct degree of smoothing by the method of generalized cross-validation. *Numerische Mathematik*, v. 31, n. 6, p. 377–403, 1979.

CURRY, B.; MORGAN, P. H. Model selection in neural networks: Some difficulties. *European Journal of Operational Research*, v. 170, p. 567–577, 2006.

CYBENKO, G. Approximation by superposition of sigmoidal functions. *Mathematics of Control, Signal and Systems*, v. 2, p. 303–314, 1989.

DAQI, G.; YAN, J. Classification methodologies of multilayer perceptrons with sigmoid activation functions. *Pattern Recognition*, v. 38, p. 1469–1482, 2005.

De Figueiredo, R. J. P. Implications and applications of Kolmogorov's superposition theorem. *IEEE Transactions on Automatic Control*, v. 25, n. 6, p. 1227–1230, 1980.

DENKER, J. et al. Large automatic learning, rule extraction, and generalization. *Complex Systems*, v. 1, p. 877–922, 1987.

DORRELL, D. G.; THOMSON, W. T.; ROACH, S. Analysis of airgap flux, current and vibration signals as a function of combination of static and dynamic airgap eccentricity in 3-phase induction motors. *IEEE Transactions on Industry Applications*, v. 33, n. 1, p. 24–34, 1997.

DRAGHICI, S. On the capabilities of neural networks using limited precision weights. *Neural Networks*, v. 15, p. 395–414, 2002.

ELKASABGY, N. M.; EASTHAM, A. R.; DAWSON, G. E. Detection of broken bars in cage rotor on an induction machine. *IEEE Transactions on Industry Applications*, v. 28, n. 1, p. 165–171, 1992.

ERDOGMUS, D. et al. Linear-least-squares initialization of multilayer perceptrons through backpropagation of the desired response. *IEEE Transactions on Neural Networks*, v. 16, n. 2, p. 325–337, 2005.

FAQ. *How Many Hidden units Should I Use?* 2006. Neural Networks FAQ Website. http://www.faqs.org/faqs/ai-faq/neural-nets/part3/section-10.html.

FILIPPETTI, F. et al. AI techniques in induction machines diagnosis including the speed ripple effect. *IEEE Transactions on Industry Applications*, v. 34, n. 1, p. 98–108, 1998.

FINNOFF, W.; HERGERT, F.; ZIMMERMANN, H. G. Improving model selection by nonconvergent methods. *Neural Networks*, v. 6, n. 6, p. 771–783, 1993.

FOGEL, D. B. An information criterion for optimal neural network selection. *IEEE Transactions on Neural Networks*, v. 2, n. 5, p. 490–497, 1991.

FUNAHASHI, K.-I. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, v. 2, n. 3, 1989.

GAO, X. M. et al. Power prediction in mobile communication systems using an optimal neural-network structure. *IEEE Transactions on Neural Networks*, v. 8, n. 6, p. 1446–1455, 1997.

GOLUB, G.; HEATH, H.; WAHBA, G. Generalized cross validation as a method for choosing a good ridge parameter. *Technometrics*, v. 21, n. 2, p. 215–224, 1979.

GORI, M.; SCARSELLI, F. Are multilayer perceptrons adequate for pattern recognition and verification? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 20, n. 11, p. 851–859, 1998.

GORMANN, R. P.; SEJNOWSKI, T. J. Analysis of hidden units in a layered network trained to classify sonar targets. *Neural Networks*, v. 1, p. 75–89, 1988.

GUTIERREZ, M.; WANG, J.; GRONDIN, R. Estimating hidden unit number for two-layer perceptrons. In: *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN'89)*. [S.l.: s.n.], 1989. p. 677–681.

HAGIWARA, K.; KUNO, K.; USUI, S. On the problem in model selection of neural network regression in overrealizable scenario. In: *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'00)*. [S.l.: s.n.], 2000. p. 461–466.

HASSIBI, B.; STORK, D. G.; WOLFF, G. J. *Optimal Brain Surgeon and General Network Pruning*. 1993. 293–299 p.

HAYASHI, M. A fast algorithm for the hidden units in a multilayer perceptron. In: *Proceedings of the 1993 International Joint Conference on Neural Networks (IJCNN'93)*. [S.l.: s.n.], 1993. p. 339–342.

HAYKIN, S. *Redes Neurais - Princípios e Prática*. 2nd. ed. [S.l.: s.n.], 2002. ISBN 85-7307-718-2.

HECHT-NIELSEN, R. Kolmogorov's mapping neural network existence theorem. In: *Proceedings of the International Conference on Neural Networks*. [S.l.: s.n.], 1987. v. 3, p. 11–14.

HECHT-NIELSEN, R. *Neurocomputing*. [S.l.: s.n.], 1990.

HENRIQUE, H. M.; LIMA, E. L.; SEBORG, D. E. Model structure determination in neural network models. *Chemical Engineering Science*, v. 55, n. 22, p. 5457–5469, 2000.

HINTON, G. E. Connectionist learning procedures. *Artificial Intelligence*, v. 40, p. 185–134, 1989.

HODGE, V. J.; AUSTIN, J. A survey of outlier detection methodologies. *Artificial Intelligence Review*, v. 22, p. 85–126, 2004.

HORNIK, K.; STINCHCOMBE, M.; WHITE, H. Multilayer feedforward networks are universal approximators. *Neural Networks*, v. 2, p. 359–366, 1989.

HSIEH, W. W. Nonlinear principal component analysis of noisy data. *Neural Networks*, v. 20, p. 434–443, 2007.

HU, X.; XU, L. Investigation on several model selection criteria for determining the number of clusters. *Neural Information Processing - Letters and Reviews*, v. 4, n. 1, p. 1–10, 2004.

HYUN, B. G.; NAM, K. Faults diagnosis of rotating machines by using neural-nets: Grnn and bpn. In: *Proceedings of the IEEE International Conference on Industrial Electronics, Control, and Instrumentation (IECON'95)*. [S.l.: s.n.], 1995. v. 2, p. 1456–1461.

ISHIKAWA, M. *A structural leaning algorithm with forgetting of link weights*. Eletrotechnical Lab., Tsukuba-City, Japan, 1990.

JOKSIMOVIé, G. M.; PENMAN, J. The detection of inter-turn short circuits in the stator windings of operating motors. *IEEE Transactions on Industrial Electronics*, v. 47, n. 5, p. 1078–1084, 2000.

KARNIN, E. D. A simple procedure for pruning back-propagation trained neural networks. *IEEE Transactions on Neural Networks*, v. 1, n. 2, p. 239–242, 1990.

KASS, R. E.; RAFTERY, A. E. Bayes factors. *Journal of the American Statistical Association*, v. 90, n. 430, p. 773–795, 1995.

KENDALL, G. D.; HALL, T. J.; NEWTON, T. J. An investigation of the generalization performance of neural networks applied to lofargram classification. *Neural Computing and Applications*, v. 1, n. 2, p. 147–159, 1993.

KERIBIN, C. Consistent estimation of the order of mixture models. *Comptes Rendus de l'Academie des Sciences - Series I:Mathematics*, v. 326, n. 2, p. 243–248, 1998.

KLIMAN, G. B. et al. Noninvasive detection of broken bars in operating induction motors. *IEEE Transactions on Energy Conversion*, v. 3, n. 4, p. 873–879, 1988.

KROSE, B.; SMAGT, P. van der. An introduction to neural networks. *fifth edn - University of Amsterdam*, 1993.

KRUSCHKE, J. K. Improving generalization in back-propagation networks with distributed bottlenecks. In: *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN'89)*. [S.l.: s.n.], 1989. p. 443–447.

KUNG, S. Y.; HWANG, J. N. An algebraic projection analysis for optimal hidden units size and learning rates in back-propagation learning. In: *Proceedings of the IEEE/INNS International Conference on Neural Networks*. [S.l.: s.n.], 1988. p. 363–370.

LAWRENCE, S.; GILES, C. L.; TSOI, A. C. *What Size Neural Network Gives Optimal Generalization? Convergence Properties of Backpropagation*. Institute for Advanced Computer Studies, University of Maryland, 1996.

LEAMER, E. E. *Specification Searches: Ad Hoc Inference with Nonexperimental Data*. [S.l.: s.n.], 1978.

LECUN, Y.; DENKER, J. S.; SOLLA, S. A. Optimal brain damage. In: TOURETZKY, D. S. (Ed.). *Advances in Neural Information Processing Systems*. [S.l.]: Morgan Kauffman, 1990. v. 2, p. 598–605.

LEE, H. K. H. Model selection for neural network classification. *Journal of Classification*, v. 18, n. 2, p. 227–243, 2001.

LEE, H. K. H. Model selection for neural network classification. *Journal of Classification*, v. 18, p. 227–243, 2001.

LEVIN, T. K. L. A. U.; MOODY, J. E. Fast pruning using principal components. In: COWAN, J. D.; TESAURO, G.; ALSPECTOR, J. (Ed.). *Advances in Neural Information Processing*. [S.l.]: Morgan Kaufmann, 1994. v. 6, p. 35–42.

LI, B. et al. Neural-network-based motor rolling bearing fault diagnosis. *IEEE Transactions on Industrial Electronics*, v. 47, n. 5, p. 1060–1069, 2000.

LI, B.; GODDU, G.; CHOW, M. Y. Knowledge based technique to enhance the performance of neural network based motor fault detectors. In: *Proceedings of the IEEE International Conference on Industrial Electronics, Control and Instrumentation (IECON'97)*. [S.l.: s.n.], 1997. v. 3, p. 1113–1118.

LI, G.; ALNUWEIRI, H.; WU, W. Acceleration of backpropagation through initial weight pre-training delta rule. In: *Proceedings of the IEEE International Conference on Neural Networks*. [S.l.: s.n.], 1993. v. 1, p. 550–585.

LOONEY, C. G. Advances in feedforward neural networks: Demystifying knowledge acquiring black boxes. *IEEE Transactions on Knowledge and Data Engineering*, v. 8, n. 2, p. 211–226, 1996.

MACKAY, D. J. C. A practical baysian framework for backprop networks. *Neural Computation*, v. 4, n. 3, p. 448–472, 1992.

MAIOROV, V.; PINKUS, A. Lower bounds for approximation by MLP neural networks. *Neurocomputing*, v. 25, p. 81–91, 1999.

MALDONADO, F. J.; MANRY, M. T. Optimal pruning of feedforward neural networks based upon the Schmidt procedure. *Proceedings of the 36th Asilomar Conference on Signals, Systems and Computers*, p. 1024–1028, 2002.

MANOLAS, S. J.; TEGOPOULOS, J. A. Analysis of squirrel cage induction motors with broken bars and rings. *IEEE Transactions on Energy Conversion*, v. 14, n. 4, p. 1300–1305, 1999.

MASAHIKO, A. Mapping abilities of three-layer neural networks. In: *Proceedings of the IEEE/INNS International Joint Conference on Neural Networks*. [S.l.: s.n.], 1989. v. 1, p. 419–423.

MASTERS, T. *Practical Neural Network Recipes in C++*. [S.l.]: Academic Press, 1993.

MEDEIROS, M. C.; TERäSVIRTA, T.; RECH, G. Building neural network models for time series: A statistical approach. *Journal of Forecasting*, v. 25, p. 49–75, 2006.

MEHROTRA, K. G.; MOHAN, C. K.; RANKA, S. Bounds on the number of samples needed for neural learning. *IEEE Transactions on Neural Networks*, v. 2, n. 6, p. 548–558, 1991.

MESHGIN-KELK, H.; MILIMONFARED, J.; TOLIYAT, H. A. Interbar currents and axial fluxes in healthy and faulty induction motors. *IEEE Transactions on Industry Applications*, v. 40, n. 1, p. 128–134, 2004.

MILIMONFARED, J. et al. A novel approach for broken rotor bar detection in cage induction motors. *IEEE Transactions on Industry Applications*, v. 35, n. 5, p. 1000–1006, 1999.

MINSKY, M. L.; PAPERT, S. A. Perceptrons. *Cambridge, MA*, 1988.

MOODY, J. E. The effective number of parameters: An analysis of generalization and regularization in nonlinear learning systems. In: LIPPMAN, D. S.; MOODY, J. E.; TOURETZKY, D. S. (Ed.). *Advances in Neural Information Processing Systems*. [S.l.]: Morgan Kaufmann, 1992. v. 4, p. 847–854.

MOODY, J. E. Prediction risk and architecture selection for neural networks. In: CHERKASSKY, V.; FRIEDMAN, J. H.; WECHSLER, H. (Ed.). *From Statistics to Neural Networks: Theory and Pattern Recognition Applications*. [S.l.]: Springer, NATO ASI Series F, 1994.

MOODY, J. E.; UTANS, J. Principled architecture selection for neural networks: Application to corporate bond rating prediction. In: LIPPMAN, D. S.; MOODY, J. E.; TOURETZKY, D. S. (Ed.). *Advances in Neural Information Processing Systems*. [S.l.]: Morgan Kaufmann, 1992. v. 4, p. 683–690.

MOZER, M.; SMOLENSKY, P. Skeletonization: A technique for trimming the fat from a network via relevance assesment. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 1989. v. 1, p. 107–115.

MULLER, G. H.; LANDY, C. F. A novel method to detect broken rotor bars in squirrel cage induction motors when interbar currents are present. *IEEE Transactions on Energy Conversion*, v. 18, n. 1, p. 71–79, 2003.

MURATA, N.; YOSHIZAWA, S.; AMARI, S.-I. Network information criterion - determining the number of hidden units for an artifitial neural network model. *IEEE Transactions on Neural Networks*, v. 5, n. 6, p. 865–872, 1994.

NAKAMURA, T. et al. A comparative study of information criteria for model selection. *International Journal of Bifurcation and Chaos*, v. 16, n. 8, p. 2153–2175, 2006.

NANDI, S.; BHARADWAJ, R. M.; TOLIYAT, H. A. Performance analysis of a three-phase induction motor under mixed eccentricity condition. *IEEE Transactions on Energy Conversion*, v. 17, n. 3, p. 392–399, 2002.

NANDI, S.; TOLIYAT, H. A. Novel frequency domain based technique to detect incipient stator inter-turn faults in induction machines. In: *Proceedings of the IEEE Conference on Industry Applications*. [S.l.: s.n.], 2000. v. 1, p. 367–374.

NANDI, S.; TOLIYAT, H. A.; LI, X. Condition monitoring and fault diagnosis of electrical machines - a review. *IEEE Transactions on Energy Conversion*, v. 20, n. 4, p. 719–729, 2005.

NATARANJAN, R. Failure identification of induction motors by sensing unbalanced stator currents. *IEEE Transactions on Energy Conversion*, v. 4, n. 4, p. 585–590, 1989.

OPPENHEIM, A. V.; SCHAFER, R. W.; BUCK, J. R. *Discrete-Time Signal Processing*. 2. ed. [S.l.]: Prentice Hall, 1999. ISBN 0-13-754920-2.

PENMAN, J.; SEDDING, H. G.; FINK, W. T. Detection and location of interturn short circuits in the stator windings of operating motors. *IEEE Transactions on Energy Conversion*, v. 9, n. 4, p. 652–658, 1994.

POGGIO, T.; GIROSI, F. Representation properties of networks: Kolmogorov's theorem is irrelevant. *Neural Computation*, v. 1, n. 4, p. 465–469, 1989.

PRECHELT, L. *Adaptive Parameter Pruning in Neural Networks*. Berkeley, CA, 1995.

PRINCIPE, J. C.; EULIANO, N. R.; LEFEBVRE, W. C. *Neural and Adaptive Systems: Fundamentals through Simulations*. [S.l.]: John Wiley & Sons, 2000.

REED, R. Pruning algorithms - a survey. *IEEE Transactions on Neural Networks*, v. 4, n. 5, p. 740–747, 1993.

RISSANEN, J. Modeling by shortest data description. *Automatica*, v. 14, p. 465–471, 1978.

RIVALS, I.; PERSONNAZ, L. A statistical procedure for determining the optimal number of hidden neurons of a neural model. In: *Proceedings of the 2nd International Symposium on Neural Computation (NC'00)*. [S.l.: s.n.], 2000.

RIVALS, I.; PERSONNAZ, L. Neural-network construction and selection in nonlinear modeling. *IEEE Transactions on Neural Networks*, v. 14, n. 4, p. 804–819, 2003.

RIVALS, I.; PERSONNAZ, L. Jacobian conditioning analysis for model validation. *Neural Computation*, v. 16, p. 401–418, 2004.

Rocha Neto, A. et al. Classificação de patologias da coluna vertebral usando redes neurais artificiais. In: *Anais do X Congresso Brasileiro de Informática em Saúde (CBIS'06)*. [S.l.: s.n.], 2006. p. 1390–1395.

ROSSI, F. et al. Mutual information for the selection of relevant variables in spectrometric nonlinear modelling. *Chemometrics and Intelligent Laboratory Systems*, v. 80, p. 215–226, 2006.

SAAD, D.; SOLLA, S. Learning from corrupted examples in multilayer networks. Aston University, n. NCRG/96/019, 1996.

SANTOS, A. H. M. et al. *Conservação de Energia: Eficiência Energética de Instalações e Equipamentos*. 2. ed. [S.l.]: Eletrobrás/PROCEL - Escola Federal de Engenharia de Itajubá-MG, 2001. ISBN 85-902115-1-7.

SCHMIDT, W. et al. Initializations, backpropagation and generalization of feed-forward classifiers. In: *Proceedings of the IEEE International Conference on Neural Networks*. [S.l.: s.n.], 1993. v. 1, p. 598–604.

SCHWARZ, G. Estimating the dimension of a model. *The Annals of Statistics*, v. 6, n. 2, p. 461–464, 1978.

SEGHOUANE, A.-K.; AMARI, S.-I. The aic criterion and symmetrizing the kullback-leibler divergence. *IEEE Transactions on Neural Networks*, v. 18, n. 1, p. 97–, 2007.

SETIONI, R.; HUI, L. C. K. Some n-bit parity problems are sovable by feedeforward netwoks with less than n hidden units. In: *Proceedings of the IEEE International Conference on Neural Networks*. [S.l.: s.n.], 1993. v. 1, p. 305–308.

SIETSMA, J.; DOW, R. J. F. Neural net pruning - why and how? In: . [S.l.: s.n.], 1988. v. 1, p. 325–333.

SOTTILE, J.; KOHLER, J. L. An on-line method to detect incipient failure of turn insulation in random-wound motors. *IEEE Transactions on Energy Conversion*, v. 8, n. 4, p. 762–768, 1993.

STAHLBERGER, A.; RIEDMILLER, M. Fast network pruning and feature extraction using the unit-obs algorithm. In: MOZER, M.; JORDAN, M.; PETSCHE, T. (Ed.). *Advances in Neural Information Processing*. [S.l.]: MIT Press, 1996. v. 9, p. 655–661.

STAVROU, A.; SEDDING, H. G.; PENMAN, J. Current monitoring for detecting inter-turn short circuits in induction motors. *IEEE Transactions on Energy Conversion*, v. 16, n. 1, p. 32–37, 2001.

STOICA, P.; SELéN, Y. Model-order selection: A review of infomation criterion rules. *IEEE Signal Processing Magazine*, p. 36–, 2004.

SWINGLER, K. *Applying Neural Networks: A Practical Guide*. [S.l.]: Academic Press, 1996.

TALLAM, R. M. et al. Neural network based on-line stator winding turn fault detection for induction motors. In: *Proceedings of the IEEE Conference on Industry Applications*. [S.l.: s.n.], 2000. v. 1, p. 375–380.

TALLAM, R. M. et al. A survey of methods for detection of stator related faults in induction machines. In: *Proceedings of the IEEE International Symposium on Diagnostics for Electric Machines, Power Electronics and Drives (SDEMPED'03)*. [S.l.: s.n.], 2003. p. 35–46.

TAMURA, S.; TATEISHI, M. Capabilities of four layered feedforward neural network: Four layers versus three. *IEEE Transactions on Neural Networks*, v. 8, n. 2, p. 251–255, 1997.

te Brake, G.; KOK, J.; VITÁNYI, P. Model selection for neural networks: comparing MDL and NIC. In: VERLEYSEN, M. (Ed.). *Proceedings of the European Symposium on Artificial Neural Networks (ESANN'94)*. [S.l.]: D-facto, 1994. p. 31–36.

TEOH, E. J.; TAN, K. C.; XIANG, C. Estimating the number of hidden neurons in a feedforward network using the singular value decomposition. *IEEE Transactions on Neural Networks*, v. 17, n. 6, p. 1623–1629, 2006.

THOMSON, W. T. On-line mcsa to diagnose shorted turns in low voltage stator windings of 3-phase induction motors prior to failure. In: *Proceedings of the IEEE International Conference on Electric Machines and Drives (IEMDC'01)*. [S.l.: s.n.], 2001. p. 891–898.

THOMSON, W. T.; BARBOUR, A. On-line current monitoring and application of finite element method to predict the level of static airgap eccentricity in three-phase induction motors. *IEEE Transactions on Energy Conversion*, v. 13, n. 4, p. 347–357, 1998.

THOMSON, W. T.; RANKIN, D.; DORRELL, D. G. On-line current monitoring to diagnose airgap eccentricity in large three-phase induction motors - industrial case histories verify the predictions. *IEEE Transactions on Energy Conversion*, v. 14, n. 4, p. 1372–1378, 1999.

VAPNIK, V. N. *Statistical Learning Theory*. [S.l.: s.n.], 1998. ISBN 0-471-03003-1.

VILLIERS, J. D.; BARNARD, E. Backpropagation neural nets with one and two hidden layers. *IEEE Transactions on Neural Networks*, v. 4, n. 1, p. 136–141, 1992.

WEIGEND, A. S.; RUMELHARD, D. E.; HUBERMAN, B. A. Generalization by weight-elimination with application to forecasting. In: LIPPMANN, R. P.; MOODY, J. E.; TOURETZKY, D. S. (Ed.). *Advances in Neural Network Processing*. [S.l.]: Morgan Kaufmann, 1991. v. 3, p. 875–882.

WIELAND, A.; LEIGHTON, R. Geometric analysis of neural network capabilities. In: *Proceedings of the IEEE International Conference on Neural Networks (ICNN'87)*. [S.l.: s.n.], 1987. v. 3, p. 385–392.

XIANG, C.; DING, S. Q.; LEE, T. H. Geometrical interpretation and architecture selection of MLP. *IEEE Transactions on Neural Networks*, v. 16, n. 1, p. 84–96, 2005.

YACOUB, M.; BENNANI, Y. Features selection and arquitecture optimization in connectionist systems. *International Journal of Neural Systems*, v. 10, n. 5, p. 379–395, 2000.

YU, X.-H. Can backpropagation error surface not have local minima? *IEEE Transactions on Neural Networks*, v. 3, n. 6, p. 1019–1021, 1992.

YUAN, H. C.; XIONG, F. L.; HUAI, X. Y. A method for estimating the number of hidden neurons in feed-forward neural networks based on information entropy. *Computers and Electronics in Agriculture*, v. 40, p. 57–64, 2003.

ZHANG, G. P. Avoiding pitfalls in neural networks research. *IEEE Transactions on Systems, Man, and Cybernetics*, C-37, n. 1, p. 3–16, 2007.