CDD:006.3

MÓDULO DE VALIDAÇÃO CRUZADA PARA TREINAMENTO DE REDES NEURAIS ARTIFICIAIS COM ALGORITMOS BACKPROPAGATION E RESILIENT PROPAGATION

CROSS VALIDATION MODULE FOR THE TRAINING OF ARTIFICIAL NEURAL NETWORKS USING BACKPROPAGATION AND RESILIENT PROPAGATION ALGORITHMS

Alaine M. Guimarães[1], Ivo M. Mathias[2], Ariangelo H. Dias[2], Jones W. Ferrari[3], Carlos R. De O. Cazelatto Junior[3].

[1] Autor para contato: Departamento de Informática, UEPG E-mail: alainemg@uepg.br
[2] Professor do Departamento de Informática, UEPG.
[3] Estudante de graduação, Bacharelado em Informática, UEPG

Recebido para publicação em 31/07/2007
Aceito para publicação em 19/11/2007

RESUMO

A metodologia de Redes Neurais Artificiais (RNAs) tem sido aplicada nas soluções de diversos problemas, dentre eles, nas aplicações voltadas a áreas específicas cujo objetivo geralmente é auxiliar na tomada de decisões. Parte destas aplicações é resolvida com simuladores, por exemplo, o JavaNNS e o SNNS. Em determinadas situações, porém, é necessário buscar informações ou valores que estão em variáveis, ou ainda, implícitos nos códigos de algoritmos de treinamento destes simuladores, não sendo acessíveis diretamente ao usuário, situação em que o uso dos simuladores torna-se insuficiente. Surge então a necessidade de desenvolver sistemas específicos, implementando todos os mecanismos de criar e de treinar as RNAs, sendo também necessário estabelecer um meio para validar os resultados obtidos dos sistemas. Diante disso, este trabalho apresenta o desenvolvimento de um Módulo de Validação Cruzada (MVC), para o Sistema Inteligente para Tratamento de Dados de Molhamento Foliar por Orvalho na Cultura do Trigo: PMNeural. No MVC, em questão, pode-se acompanhar o treinamento da rede por meio de um gráfico que exibe duas curvas, uma para o erro de treinamento e outra para o de validação. O treinamento é interrompido quando a curva de validação decresce a um erro mínimo, e antes de começar a crescer, conforme o treinamento continua.

Palavras-chave: Inteligência Artificial. Perceptron de Múltiplas Camadas. Algoritmo de Treinamento. Sistemas de Apoio à Decisão. Molhamento foliar.

ABSTRACT

Artificial neural network methodologies (ANNs) have been applied in the solution of various problems, among them those present in specific areas whose objective is usually decision making. A part of these applications is done by means of simulators, for example, JavaNNS and SNNS. In certain situations, however, it is necessary to look for information or values that are within variables, or else, implicit in training algorithms of these simulators, and thus not directly accessible to the user, a situation in which the use of simulators becomes insufficient. There arises then the necessity to develop specific systems, implementing all mechanisms for the creation and training of ANNs, while it is also necessary to establish a means for the validation of the results obtained from the systems. Therefore, this study presents the development of a Cross-Validation Module (CVM), for the Intelligent Data Processing System for the Wetting of Leaves by Dew in Wheat Plantations: PMNeural. Through this CVM one can follow the web training by means of a graphic that displays two curves, one for training errors and another for validation errors. The training is interrupted when the validation curve decreases towards a minimum error, and before it increases, as the training continues.

Keywords: Artificial Intelligence. Multi-layer Perceptron. Training Algorithm. Decision Support Systems. Foliar wetting.

1 Introdução

A Inteligência Artificial (IA) tem, como principal objetivo, representar o comportamento humano através de modelos computacionais, constituindo-se em campo de pesquisa aberto e dinâmico, tratando do estudo da solução de problemas através da distribuição de conhecimento entre diversas entidades.

Nesse sentido, as Redes Neurais Artificiais (RNAs) são sistemas computacionais, inspirados no sistema nervoso biológico. Segundo Haykin (2001), uma rede neural é um processador maciça e paralelamente distribuído, constituído de unidades de processamento simples (neurônios), que têm a propensão natural, para armazenar conhecimento experimental, tornando-o disponível para uso. Assemelha-se ao cérebro, em dois aspectos: 1- O conhecimento é adquirido pela rede, a partir de seu ambiente, através de um processo de treinamento. 2- Forças de conexão entre neurônios, conhecidas como pesos sinápticos, são utilizadas para armazenar o conhecimento adquirido. Assim, um modo de desenvolver e de implementar RNAs é utilizar simuladores, como o simulador JavaNNS (Java Neural Network Simulator), versão 1.1 (FISCHER et al., 2001), desenvolvido para ambiente Windows e Linux, baseado no SNNS (Stuttgart Neural Network Simulator), versão 4.2 (ZELL et al., 1998) que utiliza plataforma operacional Linux. Outra forma de utilizar os recursos das RNAs, é o desenvolver aplicações próprias, por meio de uma linguagem de programação, buscando solucionar um dado problema. Um exemplo deste tipo de aplicação é o PMNeural (MATHIAS, 2006), sistema computacional destinado a tratar de dados climáticos e de molhamento foliar por orvalho, visando detectar padrões de comportamento de variáveis meteorológicas, em relação ao molhamento foliar por orvalho. Esse sistema é importante no cenário agrícola, pois um dos problemas que os produtores enfrentam, para manter a produtividade das culturas, é a ocorrência de doenças, influenciadas pelas condições climáticas, com destaque para a temperatura e para a água livre sobre a superfície foliar.

Os sistemas que utilizam redes neurais buscam sua principal propriedade, a habilidade de aprender a partir de seu ambiente e, com isso, extrair conhecimento. Isso é feito através de um processo de treinamento, consistindo num processo iterativo

de ajustes, aplicando o algoritmo de treinamento a seus pesos sinápticos. Durante a fase de treinamento de RNAs, uma dificuldade é identificar um ponto de parada, de aprendizado, para a rede obter a melhor generalização e conseqüentemente o melhor resultado; sendo que uma forma de minimizá-la é utilizar o método de validação cruzada, permitindo identificar o melhor ponto de parada de uma RNA.

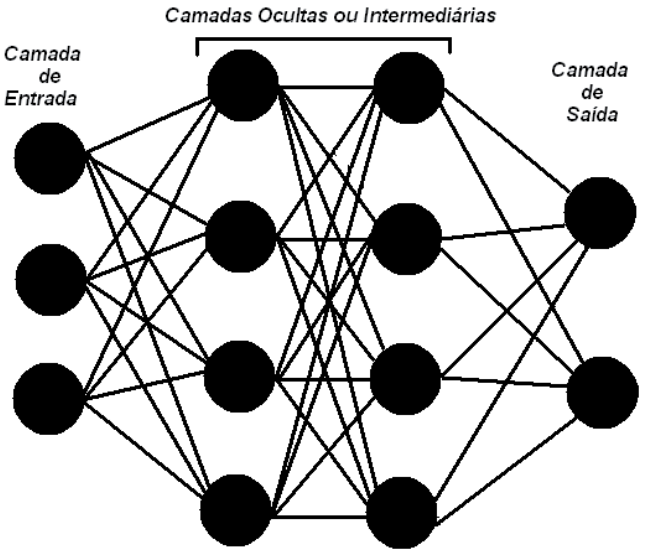
Assim, o objetivo deste trabalho foi desenvolver um Módulo de Validação Cruzada para o PMNeural, para tornar o sistema mais eficiente, já que o usuário do PMNeural tem, à sua disposição, uma ferramenta que possibilitará experimentar um número maior de arquiteturas de RNAs, em um tempo menor de processamento.

2 Materiais e métodos

2.1 Redes Neurais Artificiais

Conforme a literatura, (AZEVEDO et al., 2000; BARRETO, 1997; BRAGA et al., 2000; HAYKIN, 2001; MEDEIROS, 2003), as Redes Neurais Artificiais (RNA), ou simplesmente Redes Neurais (RN), ou ainda métodos conexionistas, compreendem uma metodologia para solucionar problemas de IA, a partir de um sistema que possui circuitos simuladores do cérebro humano, inclusive de seu comportamento, ou seja, aprendendo, errando e fazendo descobertas. São técnicas computacionais, que apresentam modelo inspirado na estrutura neural de organismos inteligentes e que adquirem conhecimento por meio da experiência.

A habilidade das RNAs em solucionar problemas complexos e variados tornou-as uma abordagem interessante, podendo ser aplicada em diversas áreas de engenharia e de ciências (SILVA et al., 2001). Os trabalhos têm sido motivados desde o começo, por reconhecerem que o cérebro humano processa informações de forma inteiramente diferente da forma do computador digital convencional. Assim, um modo de representar a estrutura genérica de uma RNA é através de um grafo direcionado, pois uma rede neural é uma estrutura de processamento de informação, distribuída paralelamente. Na representação, os neurônios correspondem aos nós deste grafo, sendo que as arestas (conexões) funcionam como as sinapses, sendo a elas atribuídos os pesos. Também é possível representar camadas, que, juntamente com os neurônios, definem a arquitetura de uma RNA: camada de entrada, camada(s) oculta(s) ou intermediária(s) e camada de saída, conforme ilustrado na Figura 1.



Fonte: Mathias (2006)

Figura 1 - RNA representada em forma de Grafo

2.2 Processo de aprendizagem

Osório e Bittencourt (2000), definem aprendizado natural como a capacidade de se adaptar, de modificar e de melhorar o comportamento e as suas respostas, sendo, pois, uma das propriedades mais importantes dos seres ditos inteligentes, sejam eles humanos, ou não. Deste modo, em se tratando de RNA, busca-se uma representação computacional deste comportamento. Haykin (2001) afirma ser a aprendizagem de RNAs, um processo em que os parâmetros livres de uma rede neural são adaptados através de um processo de estimulação pelo ambiente no qual a rede está inserida; sendo que o tipo de aprendizagem é determinado de acordo com o modo como a modificação dos parâmetros ocorre.

Assim sendo, o aprendizado conexionista é, em geral, um processo gradual e iterativo, cujos pesos são modificados várias vezes, pouco a pouco, seguindo-se uma regra de aprendizado que estabele-

ce a forma como estes pesos são alterados. O aprendizado realiza-se utilizando um conjunto de dados de aprendizado disponível (base de aprendizado ou de treinamento). O treinamento ou o aprendizado da rede ocorre em etapas, denominadas épocas ou ciclos. Esses correspondem ao número de iterações do processo, em que são aplicados todos os dados de entrada de treinamento, na rede, visando ajustar a rede, para diminuir o erro médio.

Outro aspecto relevante quanto ao aprendizado é o paradigma envolvido. Segundo Haykin (1994), existem três paradigmas diferentes, para conduzir o aprendizado, em redes neurais, sendo: supervisionado, reforço e não supervisionado. O aprendizado supervisionado, como o próprio nome sugere, é realizado sob a supervisão de um "professor" externo que tem a função de monitorar a resposta da rede, para cada entrada, conhecendo previamente a saída esperada. O aprendizado do reforço acontece, utilizando o processo de tentativa e erro, através de um "crítico". Como não há valores de saída esperados, o crítico, em vez de retornar o erro da saída da rede, retorna um sinal de reforço ou de penalidade, associado à última ação da rede. Finalmente, o aprendizado não supervisionado é baseado na auto-organização, dispensando a utilização, tanto do "crítico", quanto do "professor", não existindo o conhecimento das saídas desejadas para as entradas. Exemplos típicos que utilizam o aprendizado supervisionado são problemas de aproximar funções, de modelar sistemas e de classificar dados. O aprendizado não supervisionado aplica-se, tipicamente, a problemas de categorização de dados; enquanto que um exemplo clássico de uso de aprendizado por reforço é o de controle de um pêndulo invertido. Rohn & Mine (2003) destacam que, para aprender, não é necessário conhecer detalhadamente as relações entre as variáveis envolvidas no problema; as redes, porém, necessitam de quantidade considerável de dados históricos, para conseguirem extrair satisfatoriamente as características relevantes, existentes no conjunto de dados. Se a rede for treinada corretamente, é capaz, não somente de aproximar qualquer função, mas também de generalizar, proporcionando saídas corretas para entradas não apresentadas antes. Um modelo que tem boa generalização é o que responde corretamente aos exemplos contidos na base de aprendizado, mas também a outros exemplos, diferentes dos usados durante o processo de aprendizagem, estando contidos em uma base de teste.

Deste modo, a capacidade de generalizar é a principal capacidade, buscada nas tarefas que envolvem aprendizado; uma rede, porém, pode se especializar demasiadamente em relação aos exemplos contidos na base de aprendizado. Este tipo de comportamento gera um problema, conhecido como super-aprendizado ou super-ajuste (*over-training* / *over-fitting*). Outra situação que pode ocorrer é a rede não conseguir generalizar, gerando o problema chamado de sub-ajuste (*under-fitting*).

2.3 Algoritmos de treinamento das RNAs

Pode-se dizer que o treinamento da rede neural é o ponto onde se obtém o sucesso ou o fracasso da rede, pois, neste procedimento, submete-se a rede ao aprendizado, onde alguns fatores são relevantes, dentre eles: o algoritmo de treinamento e o número de épocas (ciclos ou iterações). Aqui, especificamente apresentamos os estudos realizados com os algoritmos de treinamento: *Backpropagation* e *Resilient Propagation*, os quais correspondem aos métodos de treinamento, implementados no sistema PMNeural com RNAs, do tipo Perceptron de Múltiplas Camadas (PMC, *Multilayer Perceptron*), que segue o paradigma de aprendizado supervisionado; sendo que, tipicamente, uma rede PMC possui uma camada de entrada, uma ou mais camadas ocultas e uma camada de saída; acrescentando ainda que, como característica básica, qualquer neurônio, em qualquer camada da rede, está conectado a todos os neurônios das camadas anteriores.

Assim, segundo Haykin (2001), as RNAs PMC têm sido aplicadas com sucesso, para resolver diversos problemas difíceis com o algoritmo de treinamento *backpropagation* (retropropagação), baseado na regra de aprendizagem por correção de erro. Na aprendizagem por correção de erro, a rede é estimulada por um vetor de entrada e sua saída desejada. A regra de ajuste consiste em achar o erro, subtraindo a resposta desejada da resposta da rede,

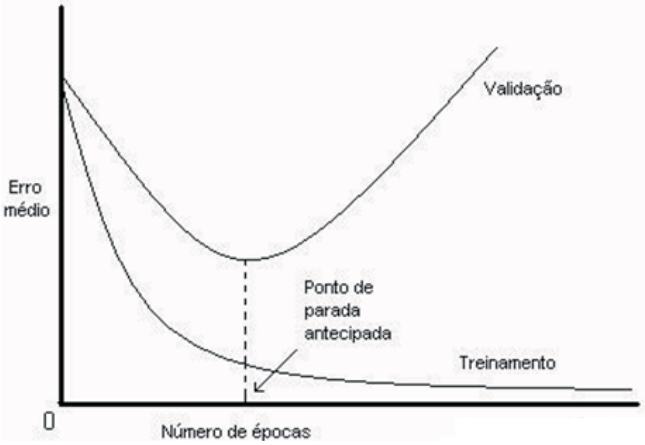
calculando o gradiente descendente da função do erro. Essa aprendizagem consiste em dois passos, através das diferentes camadas da rede: um passo para a frente, a *propagação;* e um passo para trás, a *retropropagação*. No passo para a frente, um padrão de atividade (vetor de entrada) é aplicado aos nós sensoriais da rede, sendo que seu efeito se propaga através da rede, camada por camada, produzindo um conjunto de saídas como a resposta real da rede. Nesse processo, os pesos sinápticos são mantidos fixos. Já durante o passo para trás, os pesos sinápticos são todos ajustados de acordo com uma regra de correção de erro. Assim, especificamente, a resposta real da rede é subtraída de uma resposta desejada (alvo), para produzir um sinal de erro (HAYKIN, 2001). A aplicação pode ocorrer de duas formas básicas: *seqüencial* (padrão-*standard*) e por *lote* (*batch*). Na forma *seqüencial*, o ajuste dos pesos é realizado após apresentar cada exemplo de treinamento (HAYKIN, 2001); ao contrário, na forma por *lote*, o ajuste dos pesos é realizado, após apresentar todos os exemplos de treinamento constituintes de uma época. Diante disto, o algoritmo de treinamento, *Backpropagation*, corresponde ao modo de operação seqüencial, e o algoritmo *Resilient Propagation* corresponde ao modo de ajuste dos pesos por lote.

Já o algoritmo de treinamento *Resilient Propagation* se destaca por uma característica importante, a de ser um processo que busca tornar mais eficiente o método *backpropagation*. De acordo com Riedmiller (1993), a principal adaptação foi introduzir, em cada peso, um valor de atualização individual, determinando somente o tamanho da atualização do peso; sendo esse valor adaptável, evoluindo durante o processo de aprendizagem, baseado na visão local da função de erro. Na atualização dos pesos sinápticos, independentemente do algoritmo de treinamento, um parâmetro que tem influência direta no processo de aprendizagem é a taxa de aprendizagem - um índice variando entre 0 (zero) e 1 (um), buscando determinar quanto do valor atual, da sinapse, será alterado na próxima época. Quanto menor for o parâmetro, menor serão as variações dos pesos sinápticos da rede de uma iteração para a outra, sendo mais suave a trajetória no espaço de pesos. Esta melhoria, porém, é obtida à custa de uma taxa de aprendizagem lenta. Mas se utilizarmos um parâmetro muito alto, para acelerar a taxa de aprendizagem, as grandes modificações, nos pesos resultantes, podem tornar a rede instável (oscilatória) (HAYKIN, 2001).

2.4 Validação cruzada

Uma das dificuldades do uso de RNAs consiste em identificar o melhor ponto de parada de treinamento, pois o erro de treinamento inicia com um valor alto, decresce rapidamente, e continua diminuindo lentamente, tendendo a atingir um mínimo local na superfície de erro (HAYKIN, 2001).

Assim, para identificar um ponto de parada de aprendizado, buscando obter a melhor generalização da rede, uma alternativa é utilizar a técnica da regra de parada, antecipada, com base na validação cruzada. Esta consiste em uma técnica estatística, para validar o modelo obtido durante o treinamento da rede, utilizando um conjunto de dados diferentes dos usados, para estimar os parâmetros durante o treinamento (HAYKIN, 2001). O método consiste em acompanhar a evolução do aprendizado nas curvas correspondentes aos subconjuntos de dados de treinamento e de validação (Figura 2). Deste modo, o treinamento é interrompido, quando a curva de validação decresce a um erro mínimo, e antes de começar a crescer, conforme o treinamento, continua.



Fonte: Haykin (2001)

Figura 2 - Regra de parada antecipada baseada na validação cruzada

3 Resultados

O trabalho foi desenvolvido no Laboratório InfoAgro/UEPG, do Departamento de Informática, da Universidade Estadual de Ponta Grossa, fazendo uso de dois computadores DELL Pentium 4 HT – 2.8 GHz, com 512 MB de RAM, em plataforma operacional Windows XP Professional e ambiente de Programação Borland Delphi 7 (CANTU, 2003).

Assim, o MVC foi agregado aos demais métodos do sistema PMNeural e, quando em operação, é executado em paralelo ao treinamento das RNAs, sendo que, após cada ciclo de treinamento, os pesos sinápticos do perceptron, de múltiplas camadas, deverão ser fixos; e a rede deverá operar no seu modo direto, para a frente. Neste ponto, o erro de validação é então medido para cada exemplo do subconjunto de validação. Após o final da fase de validação, o treinamento é reiniciado para novo ciclo, e o processo se repete, até que o treinamento seja interrompido, quando a curva de validação decrescer a um erro mínimo, e antes de começar a crescer.

A Figura 3 exibe os detalhes da tela padrão do PMNeural, com a qual o usuário realiza os treinamentos das RNAs. Nessa figura, pode-se observar que a interface é interativa, disponibilizando ao usuário, todas as informações necessárias ao treinamento, bem como o gráfico onde se pode acompanhar a evolução do erro e a da validação cruzada em relação ao número de épocas.

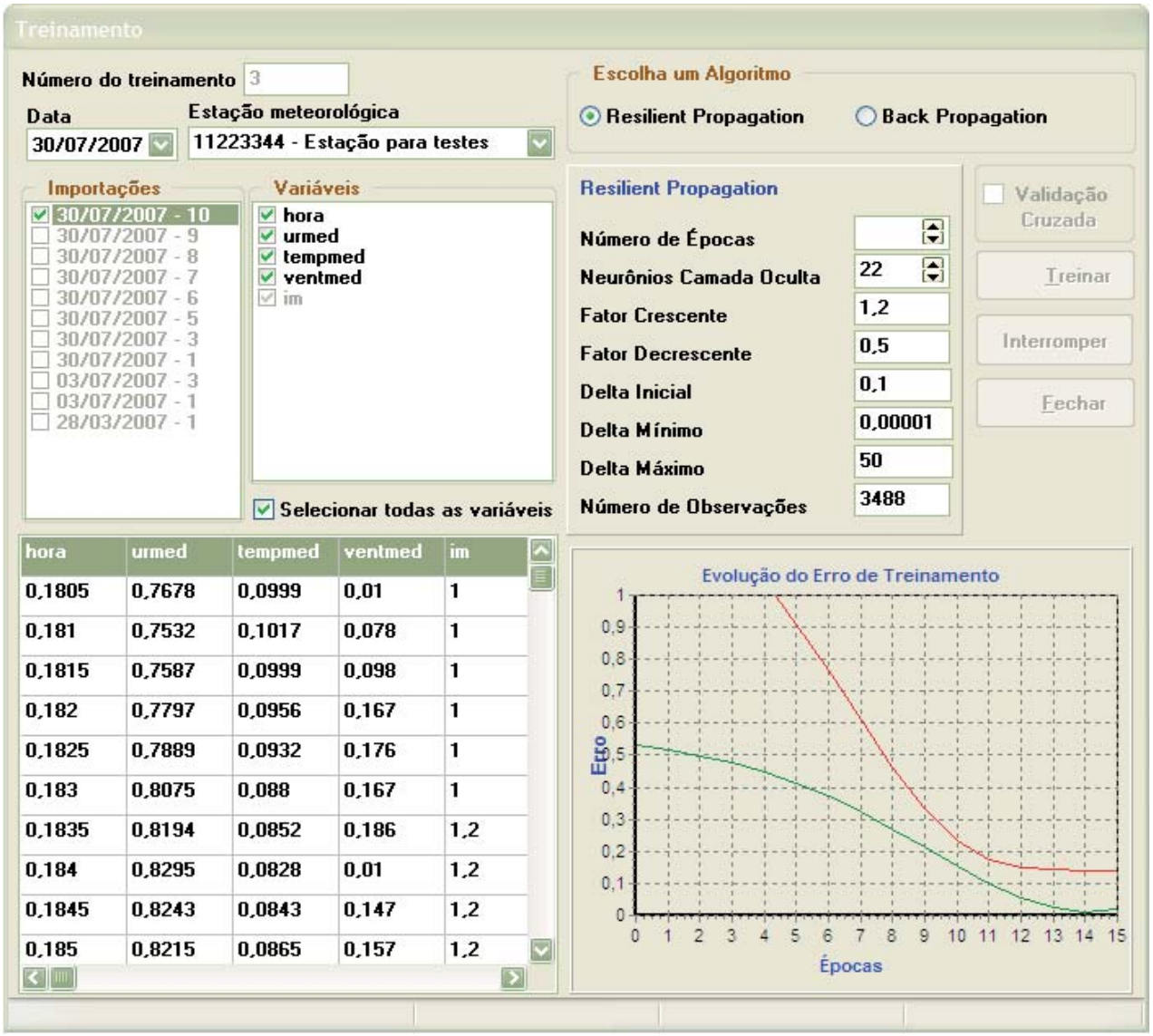


Figura 3 - Tela de treinamento - PMNeural

O procedimento inicial é definir qual a estação meteorológica a ser usada; em seguida, selecionar qual o conjunto de dados (importações) a se utilizar, para treinar. Posteriormente, deve-se escolher com quais variáveis do conjunto de dados se deseja realizar o treinamento, podendo-se optar por todas ou por algumas especificamente; sendo que, ao se proceder à escolha, define-se o número de neurônios da camada de entrada da RNA.

O passo seguinte é escolher o algoritmo de treinamento: *Resilient Propagation* ou *Back Propagation, sabendo-se que, p*ara todos os parâmetros do processo de treinamento, existem valores recomendados, mas podem ser alterados pelo usuário. O número de épocas era o parâmetro que exigia maior atenção do usuário, pois o número de iterações é um dos itens que determinam o sucesso ou o insucesso do treinamento, da rede neural, em relação a um determinado conjunto de dados, ou seja, a convergência da rede para o melhor resultado possível. Com a implantação do MVC, o parâmetro número de épocas passa a ser automático, embora o usuário possa alterá-lo; sendo que o sistema inicia com um valor, calculado em função do número de neurônios da camada oculta, multiplicado por cem. Por exemplo, se na rede que está sendo projetada, são três neurônios de entrada, onze neurônios da camada oculta e dois neurônios de saída, o cálculo para determinar o número de épocas é 11 * 100 = 1100. Este procedimento é empírico, sendo que o método adotado, neste trabalho, não corresponde a uma técnica padrão, nem a um modo que tenha a sua eficiência provada matematicamente. O número de observações é um valor fornecido automaticamente pelo sistema, conforme a quantidade de registros existentes no arquivo de importação correspondente.

O próximo requisito, para o treinamento, é construir a RNA, que se obtém, clicando-se no botão ***Treinar***, que então, emite mensagem, solicitando confirmação, se o usuário realmente deseja iniciar o treinamento da rede. O tempo de treinamento das redes é influenciado diretamente pelo número de épocas, pelo número de observações e pelo número de neurônios nas diversas camadas da RNA; assim como, pela capacidade para o computador, em uso, processar, e pela quantidade de memória RAM. Sendo assim, o tempo de treinamento de uma RNA pode variar entre segundos, minutos e horas. Esse tempo é um dos aspectos que será diretamente influenciado pelo funcionamento do MVC, pois o treinamento será encerrado, quando a curva de validação decrescer a um erro mínimo, e antes de começar a crescer; sendo esta uma situação que possivelmente abreviará o treinamento das RNAs. Uma vez que o treinamento seja realizado com sucesso, um relatório, como o da Figura 4 é exibido.

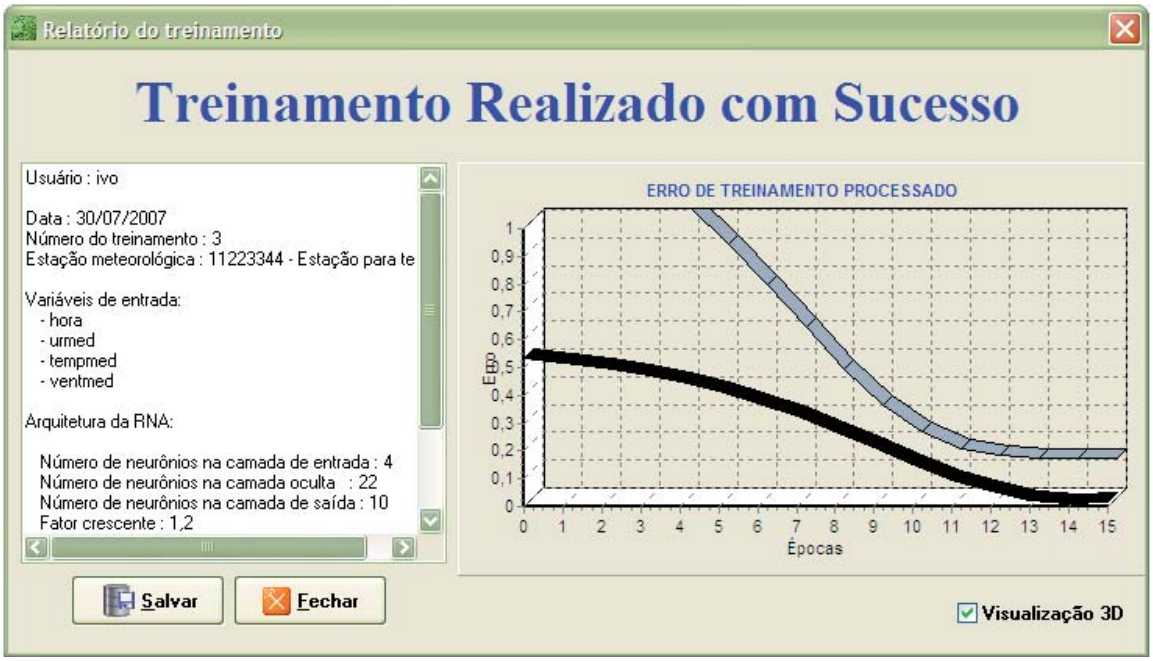


Figura 4 - Tela relatório de treinamento - PMNeural

Neste relatório, são destacados todos os parâmetros utilizados no treinamento, bem como o gráfico da evolução do treinamento e da validação. A partir do treinamento em que o usuário considere satisfatório, o erro obtido, o relatório pode ser gravado em disco, bastando, para isso, clicar no botão ***Salvar***. Este procedimento se constitui no armazenamento da RNA, treinada no banco de dados do sistema.

Conclusões

A utilização do Módulo de Validação Cruzada, no PMNeural, torna o sistema mais eficiente, pois, anteriormente, a tarefa de identificar o melhor ponto de parada de aprendizado era realizada empiricamente; agora, porém, o usuário do PMNeural tem, à sua disposição, uma ferramenta que possibilita experimentar um número maior de arquiteturas de RNAs, em um tempo menor de processamento.

Para as redes do PMNeural poderem gerar resultados satisfatórios, porém, é necessário os dados, utilizados nos treinamentos das RNAs, possuírem características relevantes, para os algoritmos de treinamento poderem extrair padrões de comportamento das variáveis meteorológicas, em relação ao molhamento foliar por orvalho.

REFERÊNCIAS

AZEVEDO, F. M.; BRASIL, L. M. e OLIVEIRA, R. C. L. de. **Redes neurais com aplicações em controle e em sistemas especialistas**. Florianópolis: Visual Books Editora, 2000, 401p.

BARRETO, J. M. Introdução às redes neurais artificiais, In: ESCOLA REGIONAL DE INFORMÁTICA DA SBC REGIONAL SUL, 5, 1997, Florianópolis. **Anais**... Florianópolis, Maringá, 1997, p. 41-71.

BRAGA, A. P.; CARVALHO, A. C. P. L. F.; LUDERMIR, T. B. **Redes neurais artificiais: teoria e aplicações.** Rio de Janeiro, LTC – Livros Técnicos e Científicos, 2000, 262p.

CANTU, M. **Dominando o Delphi 7, a bíblia**. Tradução de Kátia Aparecida Roque. São Paulo: Pearson Education do Brasil, 2003.

FISCHER, I., et. al. **JavaNNS –** Java Neural Network Simulator**:** user manual – version 1.1. University of Tübingen, Wilhelm-Schickard - Institute for Computer Science, Department of Computer Architecture, 2001. Disponível em: http://www-ra.informatik.uni-tuebingen.de/ software/JavaNNS/manual/JavaNNS-manual.pdf Acesso em: setembro 2004.

HAYKIN, S. **Redes neurais princípios e prática**. Porto Alegre: Bookman, 2001, 900p.

HAYKIN, Simon**. Neural netwoks :** a comprehensive foundation**.** New York: Macmillan College Publishing, 1994.

MATHIAS, Ivo M. **Aplicação de redes neurais artificiais na análise de dados de molhamento foliar por orvalho**. Botucatu, 2006. 120 p. Tese (Doutorado em Agronomia/Energia na Agricultura) Universidade Estadual Paulista - Faculdade de Ciências Agronômicas, São Paulo.

MEDEIROS, L. F. de. **Redes neurais em Delphi**. Florianópolis: Visual Books Editora, 2003, 115p.

OSÓRIO, F. S.; BITTENCOURT, J. R. Sistemas Inteligentes baseados em redes neurais artificiais aplicados ao processamento de imagens, In: WORKSHOP DE INTELIGÊNCIA ARTIFICIAL, 2000. **Apostila-seminário**, Santa Cruz do Sul, UNISC – Universidade de Santa Cruz do Sul - Departamento de Informática, 2000.

RIEDMILLLER, M.; BRAUN, H. A direct adaptive method for faster backpropagation learning: the RPROP algorithm, In: PROCEEDINGS OF THE IEEE, INTERNATIONAL CONFERENCE ON NEURAL NETWORKS, IEEE. **Anais...** Press, New York, 1993, p. 586-591

ROHN, M. da C.; MINE, M. R. M. Uma aplicação de redes neurais artificiais à previsão de chuvas de curtíssimo prazo. In: SIMPÓSIO BRASILEIRO DE RECURSOS HÍDRICOS, 15. 2003. **Anais....** Curitiba, Associação Brasileira de Recursos Hídricos, 2003, CD-ROM.

SILVA, I. N. da. et. al. Projeto e análise de uma rede neural para resolver problemas de programação dinâmica. **Sba Controle & Automação**, v.12, n.1, 2001, p.1-10.

ZELL, A. et al. SNNS - Stuttgart Neural Network Simulator. **User Manual - Version 4.2**. University of Stuttgart, Institute for Parallel and Distributed High Performance Systems, 1998, Disponível em: http://www-ra.informatik.uni-tuebingen.de/downloads/SNNS/SNNSv4.2 Acesso em: setembro 2004.