Rede Neuronal Artificial para o Reconhecimento de Caracteres com Interface Gráfica

João Paulo Teixeira*, José Batista*, Anildio Toca**, João Gonçalves**, e Filipe Pereira**
* Departamento de Electrotecnia
Escola Superior de Tecnologia e de Gestão - Instituto Politécnico de Bragança
Campus Sta. Apolónia – Bragança
Telf: +351 273 303 3129; fax: +351 273 313 051; e-mail: joaopt@ipb.pt, jbatista@ipb.pt

**Escola Superior de Tecnologia e de Gestão - Instituto Politécnico de Bragança
Campus Sta. Apolónia – Bragança
Telf: +351 273 303 3129; fax: +351 273 313 051; e-mail: anildio.cb.toca@alunos.ipb.pt, joao.nc.goncalves@alunos.ipb.pt, filipe.gp.pereira@alunos.ipb.pt

Resumo **– Os propósitos deste trabalho são meramente académicos e visam a experimentação das Redes Neuronais Artificiais numa situação típica de reconhecimento de caracteres. Adicionalmente foi experimentada a utilização concomitante do LabVIEW e do Matlab, tirando partido da interface gráfica do LabVIEW e da facilidade de utilização das RNA no Matlab. Os caracteres reconhecidos são os números de 0 a 9 e as letras maiúsculas de A a Z. Cada símbolo é codificado numa matriz booleana de 5×5. Como existem 36 caracteres no total, resulta uma matriz de 25×36 (dados de entrada da rede). Para a correspondência de cada vector de entrada, é usada uma matriz de saída de 36×36 que toma valores '0' ou '1'. Para a RNA foi adoptada uma tipologia *feedforward* com duas camadas escondidas, funções de activação tansig, e trainrp como função de treino [1]. Para o treino da rede recorreu-se à Neural Network Toolbox™ do Matlab®, e ao LabVIEW™ para o desenvolvimento da interface gráfica. Deste modo, resultou um ambiente gráfico muito amigável e de grande utilidade, devido à troca de dados entre estes dois ambientes de programação. A rede apresenta uma boa tolerância a falhas e bom desempenho.**

Palavras-chave: Matlab®, LabVIEW™, Rede Neuronal Artificial, Reconhecimento de Caracteres.

1. Introdução

As Redes Neuronais Artificiais (RNAs) são técnicas computacionais que apresentam um modelo inspirado na estrutura do cérebro humano, com a particularidade de adquirirem conhecimento através da experiência. Apresentam características únicas, que não se encontram noutros mecanismos, o que tornam a sua utilização apetecível, sendo de destacar [2]:
A capacidade de aprendizagem e generalização, a capacidade de processamento paralelo, a sua aplicabilidade a problemas não lineares, a adaptabilidade a novos casos, robustez, e flexibilidade. O reconhecimento automático de caracteres é um tema actual e de grande interesse. Devido às suas características, as RNAs têm dado um grande contributo nesta área. O crescente aumento da capacidade de processamento dos computadores pessoais comerciais, aliado ao desenvolvimento de ferramentas computacionais adequadas e poderosas, proporcionam o rápido desenvolvimento de variadas aplicações, incluindo por vezes processamentos computacionais bastante complexos. Fazem parte destas ferramentas o Matlab® (MATrix LABoratory) e o LabVIEW™ (Laboratory Virtual Instrument Engineering Workbench), softwares de desenvolvimento das empresas Mathworks™ (www.mathwoks.com) e National Instruments™ (www.ni.com), respectivamente.
Neste artigo é apresentado o resultado de uma rede neuronal aplicada ao reconhecimento de caracteres, com uma interface gráfica muito útil e amigável para proceder a testes da mesma. Para a concepção e treino da RNA recorreu-se ao Matlab (ver. R2008a) em conjunto com a *toolbox* de redes neuronais (ver. 6.0) deste software. Na versão anterior do Matlab, a sintaxe das funções utilizadas e apresentadas neste artigo, seria ligeiramente diferente. A interface gráfica foi desenvolvida em LabVIEW (ver. 8.5).
O LabVIEW é uma linguagem de programação gráfica, bastante utilizada na indústria e no ensino, tendo sido inicialmente usada para visualização de medidas instrumentais de laboratório, funcionando como instrumento de medida virtual. As primeiras aplicações desenvolvidas com esta linguagem surgiram por volta de 1986, e desde então ganhou muita popularidade, devido principalmente à sua eficiência amplamente demonstrada, fácil utilização, desenvolvimento rápido de aplicações e possibilidade de reutilização de código [3], [4]. A sua natureza gráfica torna esta linguagem ideal para aplicações de T&M (Teste e Medida), automação, controlo de instrumentos, aquisição e análise de dados, e outras aplicações na área da Engenharia. Para criar programas, em vez de linhas de texto, esta linguagem utiliza ícones ligados entre si, e uma notação de diagrama de blocos facilmente inteligível e muito utilizada em actividades de I&D (Investigação e Desenvolvimento). Em LabVIEW um programa designa-se por VI (*Virtual Instrument*), sendo constituído por dois componentes essenciais: Painel Frontal e Diagrama. O Painel Frontal, contém entre outros, elementos de controlo, indicadores, gráficos e tabelas. O

Diagrama corresponde ao código do programa, incluindo, por exemplo, funções, rotinas, estruturas, ciclos e fórmulas. Por motivos de eficiência, e sobretudo quando as aplicações são complexas, é usual organizar o programa num VI principal e Sub-VIs ligados entre si, resultando uma arquitectura hierárquica da aplicação. Uma característica interessante do LabVIEW, aproveitada no âmbito deste trabalho, é a possibilidade de este permitir executar *.m files scripts do Matlab. Desta forma podem ser trocados dados entre estes dois ambientes de programação, aproveitando características gráficas do LabVIEW e código desenvolvido em Matlab.

2. Descrição do Sistema Desenvolvido

Na figura 1 é apresentado um diagrama de blocos genérico do sistema desenvolvido para o reconhecimento de caracteres com interface gráfico, utilizando os dois ambientes de programação referidos.

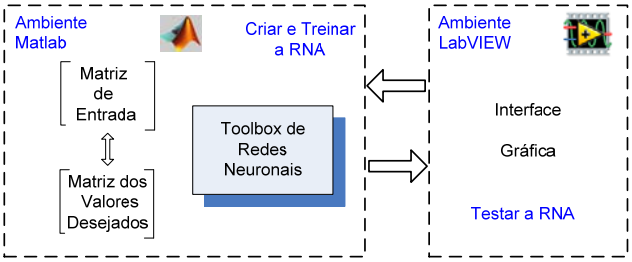


Fig. 1. Diagrama de blocos do sistema desenvolvido

Os procedimentos principais em Matlab são a codificação dos dados para apresentar à RNA, a concepção e o treino da mesma. Em LabVIEW, o procedimento principal consiste no teste da RNA com interacção do utilizador, efectuada no ambiente gráfico implementado para o efeito.

A. Definição dos padrões de entrada - Matrizes

Na figura 2 é mostrada a forma como são codificados alguns caracteres e a forma como é criada a 1.ª coluna da matriz de entrada para a rede (neste caso correspondente ao caractere zero).

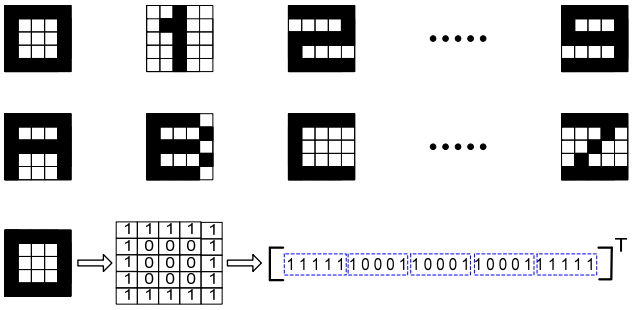


Fig. 2. Codificação dos caracteres

Cada caractere é codificado numa matriz do tipo booleana de 5×5. O valor '1' corresponde ao ponto marcado na matriz como *true* e o valor '0' ao ponto marcado como *false*, sendo portanto, cada padrão definido por 25 bits. Esta matriz é posteriormente transformada numa matriz coluna de dimensão 25×1. Codificados todos os caracteres desta forma, é criada a matriz de entrada para apresentar à rede neuronal, de dimensão 25×36, pois no total são codificados 36 caracteres.
Em Matlab é utilizada uma estrutura do tipo array de células com matrizes e a função reshape() para concretizar estes procedimentos. A correspondência entre a matriz de entrada e a matriz com os valores desejados (*Target*) é feita atribuindo o valor '1' na coluna que se pretende para identificar o caractere que foi previamente codificado na matriz de entrada. Este processo, definição das matrizes, é ilustrado na figura 3.

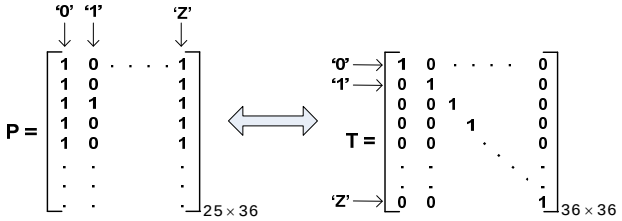


Fig. 3. Matrizes para a RNA

Conforme ilustra a figura, a matriz P contém os dados de entrada para o treino da RNA, com todos os caracteres codificados. A matriz T (*Target*) contém a codificação para a identificação pretendida dos caracteres. Na prática, esta é, uma matriz identidade de tamanho 36×36.

B. Tipologia, arquitectura e treino da RNA

A rede criada é do tipo FF (*FeedForward*), com duas camadas escondidas (intermédias) e treinada com o algoritmo *backpropagation*. As redes FF são estáticas, não têm elementos de realimentação nem atrasos, sendo a saída calculada directamente a partir da entrada através de conexões do tipo *feedforward*. *Backpropagation* é um algoritmo de gradiente descendente, tal como a regra de Widrow-Hoff, em que os pesos se movem na direcção do gradiente negativo da função performance [1]. Uma componente fundamental das redes neuronais é a função de activação para limitar a amplitude da saída do neurónio e introduzir uma componente de não linearidade no processo computacional. Neste trabalho, desenvolveu-se para o efeito uma *.m file, sendo criada e treinada uma RNA, recorrendo a uma função especifica da biblioteca de redes neuronais do Matlab, com a seguinte sintaxe simplificada:

net=newff(P,T,[40 36],{'tansig','tansig','tansig'},'trainrp');

Onde P representa a matriz de dados de entrada e T a matriz *target*. O número de nós das camadas escondidas é 40 e 36, respectivamente da 1.ª e 2.ª camada escondidas. O número de nós da camada de entrada é definido pelo número de linhas da matriz P, sendo então 25 nós. O número de nós da camada de saída é determinado pelo número de linhas da matriz T, portanto 36 neste caso. A função de transferência adoptada para as três camadas da rede é a tansig (Tan-Sigmoid), enquanto que a função de treino escolhida é a trainrp (*resilient backpropagation*). Todos os outros parâmetros da função não referidos, são os que vêm por defeito no Matlab. Desta forma, é criada uma

rede *feedforward backpropagation*, com duas camadas escondidas e uma camada de saída.
A utilização da função tansig na camada de saída permite obter valores entre 0 e 1 na saída, o que é conveniente no âmbito deste trabalho A função de treino trainrp implementa o algoritmo mais rápido para problemas de reconhecimento de padrões, sendo os requisitos de memória para este algoritmo relativamente inferiores em comparação com outros métodos [1].
Para treinar a rede, em Matlab, é utilizada a função train(), com a seguinte sintaxe simplificada:

net=train(net,P,T);

Sendo o argumento net a rede criada pela função newff().
A versão da toolbox de redes neuronais utilizada, disponibiliza uma ferramenta muito útil, o GUI (*Graphical User Interface*) - nntraintool. Nesta janela, pode ser visualizada toda a actividade da rede na fase de treino, incluindo um esquema gráfico da arquitectura, o nome dos algoritmos utilizados, o n.º de iterações efectuadas, o tempo, a performance, o gradiente, e visualização de gráficos. A figura 4 mostra a janela parcial deste GUI, para a rede criada.

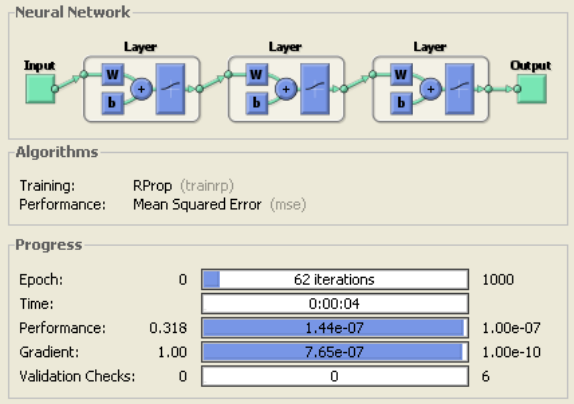


Fig. 4. Janela parcial do GUI nntraintool

A performance da rede treinada pode ser visualizada num gráfico que também pode ser obtido a partir desta ferramenta. Trata-se de um gráfico que representa a curva Épocas vs MSE (*Mean Square Error*). O valor do erro médio quadrático utilizado foi de 10e-08. No treino da rede foram programadas 1000 épocas (iterações), tendo sido registadas 62 iterações, uma vez que aqui foi atingido o MSE mínimo. Alguns resultados obtidos com esta rede são apresentados na secção 3.

C. Teste da RNA – Simulação

Depois de treinada e analisada a performance da rede, esta está pronta para ser testada. Este teste consiste em verificar como a rede classifica determinados padrões introduzidos na sua entrada que não foram utilizados no treino. Trata-se de testar o comportamento da rede em relação à sua capacidade de generalização e tolerância a falhas. Em Matlab, estes testes são realizados fazendo a simulação da rede, usando para o efeito a função sim() com a seguinte sintaxe simplificada:

y=sim(net,P');

Onde y é o resultado devolvido pela função, net é a rede treinada e P' é a matriz de entrada. Esta matriz P´ corresponde ao caractere que se pretende reconhecer, e será classificado segundo as regras de aprendizagem da rede. O resultado devolvido pela função sim(), neste caso uma matriz de dimensão 36×1, contém em cada índice, um valor entre 0 e 1. A 1.ª posição da matriz corresponde ao caractere ´0´, a 2.ª posição ao caractere ´1´, e assim sucessivamente, até á posição 36 que corresponde ao caractere ´Z´. Para a classificação do padrão de entrada, é procurado nesta matriz de saída, o valor numérico mais próximo de 1. Deste modo, qualquer padrão introduzido na matriz de entrada é reconhecido sempre como um dos caracteres previamente codificados, com uma determinada probabilidade de certeza. Alguns destes valores obtidos nos testes efectuados são apresentados na secção 3.

D. Interface gráfica

Para tornar prático, funcional e agradável o teste da rede treinada, foi desenvolvida uma interface gráfica em LabVIEW. A janela do Painel Frontal desta interface é mostrada na figura 5.

Fig. 5. Painel Frontal da interface gráfica em LabVIEW

Na parte superior da janela aparece um controlo do tipo *string* com a *label* "Nome File Dados". Este controlo destina-se a seleccionar o nome da rede que previamente foi treinada em Matlab. Deste modo, podem ser treinadas várias redes com diferentes parâmetros, e neste ambiente gráfico podem ser testadas todas e comparados os resultados rapidamente. Para tal basta mudar o nome da rede previamente gravada em Matlab, e pressionar o botão "Executar". A *string* com a *label* "Path de Trabalho" indica apenas o caminho do directório de trabalho. A matriz booleana de entrada com a *label* "Matriz de Teste" destina-se à introdução do novo padrão a ser classificado. Esta matriz é transformada numa matriz com valores numéricos, sendo um dos parâmetros de entrada para o Matlab. O resultado devolvido pelo Matlab (depois de aplicado o critério de classificação referido na secção anterior), é mostrado na matriz de saída "Caractere

Reconhecido". No exemplo mostrado na figura 5, pode observar-se que o padrão introduzido é reconhecido como sendo o caractere ´1´, sendo que em relação ao codificado, este apresenta um nível de ruído com 7 pontos errados na matriz de entrada. Na parte inferior da janela do Painel Frontal, referenciada como "Caracteres Codificados para Treino da Rede", existem dois controlos para implementar menus, designados em LabVIEW por *Ring Controls*, com as *labels* "Números" e "Letras". Quando se selecciona nestes menus o caractere desejado, automaticamente aparece a respectiva codificação na matriz booleana situada abaixo destes menus. O objectivo desta funcionalidade, é saber exactamente em qualquer momento, o modo como se codificaram todos os caracteres em causa (números e letras), conjugando assim, esta informação com o resultado obtido na simulação da RNA enquanto se efectuam os testes. Nesta zona inferior da janela, surge ainda uma caixa com a *label* "Erro" que se destina à gestão de eventuais erros de execução do programa que possam ocorrer. Parte do código (diagrama em LabVIEW) correspondente ao Painel Frontal descrito é mostrada na figura 6, designadamente a parte que trata da troca de dados entre LabVIEW e Matlab.

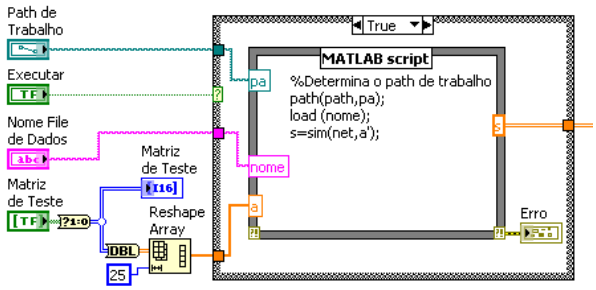


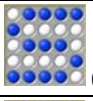
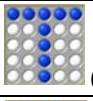
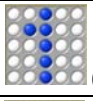
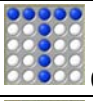
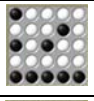
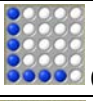
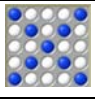
Fig. 6. Diagrama parcial do Painel Frontal

As linhas de texto que se encontram dentro da estrutura da "MATLAB script" correspondem ao código em Matlab executado nesse script. Na parte esquerda da estrutura aparecem as variáveis de entrada, e na parte direita as variáveis de saída. De notar que a variável de saída "s" é a matriz devolvida pela função sim() do Matlab. Todos os procedimentos, dentro desta estrutura, são executados sempre que é pressionado o botão "Executar" do Painel Frontal. Quando a aplicação principal desenvolvida em LabVIEW é executada, surge a janela mostrada na figura 5, e automaticamente é aberta a janela de comandos do Matlab (sem o *layout* normal). Isto significa que não é necessário ter o Matlab aberto para executar a aplicação desenvolvida.

3. Resultados Experimentais

Neste trabalho, foi utilizada uma matriz de 5×5 para a codificação de 36 caracteres (números de 0 a 9 e letras de A a Z). A RNA foi treinada com todos os caracteres codificados sem ruído. A tabela I apresenta resultados de vários testes realizados, para a rede criada e treinada conforme descrito na secção 2.

TABELA I
ALGUNS RESULTADOS EXPERIMENTAIS

# Bits Errados	Padrão de Entrada	Resultado da RNA
0		 (2); 0.99
4		 (3); 0.98
3		 (3); 0.83
4		 (4); 0.88
3		 (5); 0.97
2		 (9); 0.15
1		 (6); 0.44
1		 (G); 0.38
3		 (8); 0.93
4		 (7); 0.79
3		 (A); 0.55
2		 (A); 0.95
5		 (S); 0.79
4		 (T); 0.28
5		 (1); 0.62
4		 (T); 0.22
4		 (L); 0.92
6		 (X); 0.79

Nesta tabela, podem ser visualizados vários padrões aleatórios introduzidos na matriz de entrada, com alguns bits errados em relação ao resultado expectável, e o resultado devolvido pela RNA para o respectivo padrão. A maneira como todos os caracteres foram codificados pode ser rapidamente visualizada no Painel Frontal da interface gráfica desenvolvida para esta aplicação. Quando o padrão da matriz de entrada é um caractere codificado sem qualquer ruído, a RNA identifica sem qualquer dúvida, como era de esperar, o caractere exacto. Neste caso, a matriz devolvida pela RNA apresenta sempre o valor 1 (ou muito próximo) no índice correspondente ao caractere em causa, e sempre o valor 0 (ou muito próximo) nos outros casos. Na tabela I também é apresentado o valor numérico que a rede devolve no processo de reconhecimento do caractere em causa, sendo este valor indicado com duas casas decimais. Este número determina a classificação do respectivo caractere, sendo o critério utilizado a correspondência ao valor mais próximo de 1, como referido na subsecção C do ponto 2.

De forma a tornar mais realista o tema abordado neste artigo, o conjunto de dados que formam as matrizes de entrada/saída usado no treino poderia incluir algum ruído. Por outro lado, um dos objectivos do trabalho poderia ser a optimização da rede criada, nomeadamente em relação ao número de nós nas camadas escondidas. Existem trabalhos realizados neste sentido [5], em que na codificação dos caracteres é utilizada, por exemplo, uma matriz de 10×10, sendo cada um dos caracteres codificado com 5 matrizes com ruído e 5 matrizes sem ruído. Deste modo, resultariam na matriz de entrada, 10 colunas para cada caractere.

4. Conclusões

Este artigo descreve o desenvolvimento de uma RNA para o reconhecimento de caracteres, com interface gráfica para visualizar a matriz de entrada e o caractere identificado pela rede. A arquitectura e treino da rede foram desenvolvidos em Matlab, e a interface gráfica em LabVIEW.

Demonstrou-se que para o reconhecimento de caracteres, através dos testes realizados, a RNA implementada, do tipo *feedforward* com duas camadas intermédias, funções de activação tansig e função de treino trainrp, apesar de ser utilizada uma matriz de pequena dimensão (5×5) para a codificação dos dígitos, conseguiram-se bons resultados.

Constatou-se que a *toolbox* de redes neuronais do Matlab é uma poderosa ferramenta permitindo um rápido desenvolvimento deste tipo de redes.

A utilização de *.m files em conjunto com a ferramenta nntraintool disponibilizada no Matlab, também facilitam estas tarefas. A versão utilizada foi a R2008a com *toolbox* actualizada.

Para o teste da RNA, no tipo de aplicação apresentada neste artigo, é muito importante ter uma interface gráfica boa com o utilizador. Recorrendo ao LabVIEW para implementar esta interface, consegue-se de facto um ambiente muito amigável, funcional, útil, fácil de trabalhar e intuitivo. Fica assim demonstrado, que conjugando o LabVIEW com scripts do Matlab é possível desenvolver aplicações muito interessantes para a resolução de problemas em várias áreas. Um dos pontos fortes desta estratégia assenta na simplicidade e rápido desenvolvimento da aplicação, sendo quanto a nós, a utilização do LabVIEW para o desenvolvimento da interface gráfica, uma vantagem em relação à estratégia seguida em [6], uma vez que nesse trabalho a interface gráfica foi desenvolvida utilizando o GUIDE (Graphical User Interface Development Environment) do Matlab.

Em comparação com os resultados obtidos em [6], verifica-se que a RNA com duas camadas escondidas poderá ter algumas vantagens para esta aplicação, nomeadamente em relação à capacidade de generalização da rede.

Referências

[1] Howard Demuth, Mark Beale, Martin Hagan “Neural Network Toolbox™6, User’s Guide”, The Mathworks™, release 2008a, March 2008.

[2] Paulo Cortez e José Neves, “Redes Neuronais Artificiais”, Unidade de Ensino, Departamento de Informática, Escola de Engenharia, Universidade do Minho, Braga, Portugal 2000.

[3] Gary W. Johnson – LabVIEW Garaphical Programming, Pratical Applications In Instrumentation and Control, 2nd Edition, McGraw, 1997.

[4] Rahman Jamal, Herbert Pichlik - LabVIEW Applications and Solutions, National Instruments, Virtual Instrumentation Series; Prentice Hall; illustrated edition (22 Dec. 1998).

[5] Jearanaitanakij K., Pinngern O., “Hidden Unit Reduction of Artificial Neural Network on English Capital Letter Recognition,” Cybernetics and Intelligent Systems, 2006 *IEEE Conference On*, 7-8 June 2006.

[6] João Paulo Teixeira, José Batista, Anildio Toca, João Gonçalves, Filipe Pereira, “Identificação de Caracteres com Rede Neuronal Artificial com Interface Gráfica”, Engenharia`2009 – Inovação e Desenvolvimento, 5.ª Conferência de Engenharia, UBI, Covilhã, 25 a 27 Novembro de 2009.