# A implementação e o estudo de redes neurais artificiais em ferramentas de *software* comerciais

Cleber Gustavo Dias
Professor do Departamento de Ciências Exatas – Uninove.
São Paulo – SP [Brasil]
diascg@uninove.br

Este artigo apresenta uma abordagem da implementação de redes neurais artificiais em ferramentas de *software* comerciais, possibilitando o estudo e a avaliação de sistemas inteligentes voltados para fins acadêmicos, ou ainda, para profissionais que atuam em diversas áreas do conhecimento, auxiliando-os no processo de tomada de decisão. A metodologia de implementação de uma rede neural em ferramenta de *software* como o Microsoft Excel® e os seus resultados são apresentados neste trabalho.

Palavras-chave: Inteligência artificial. Redes neurais artificiais. Sistemas de informação.

## 1 Introdução

O campo da inteligência artificial evoluiu nas últimas décadas em razão do avanço acentuado dos componentes eletrônicos, da área da ciência da computação e principalmente pelo desenvolvimento de novas ferramentas de *software*.

Os fatos e acontecimentos remetem à história de descobertas, desconfianças e desafios, nos quais a área da inteligência artificial esteve envolvida (RUSSEL; NORVIG, 1995). Sabe-se que atualmente os sistemas inteligentes são capazes de interferir em um processo de tomada de decisão, de forma abrangente e, ao mesmo tempo, eficiente. Os sistemas de informação na sociedade moderna contemplam o uso de agentes inteligentes, oferecendo confiabilidade e maior competitividade (PLANTULLO, 2002).

As principais técnicas de inteligência artificial, como as redes neurais artificiais (HAYKIN, 2001), a lógica *fuzzy* e os algoritmos genéticos, são aplicadas atualmente em grande variedade de processos, sistemas e dispositivos. Implementações de redes neurais artificiais que percorrem desde a área da economia e mercado financeiro até o diagnóstico complexo de falhas em máquinas e/ou processos industriais estão disponíveis para inúmeros profissionais a fim de auxiliá-los na avaliação de possíveis problemas (PALADE; BOCANIALA; JAIN, 2006; AZEVEDO; BRASIL; OLIVEIRA, 2000; AQUINO; CARVALHO; SOUZA, 1999).

Trabalhos no campo da automação industrial e robótica apresentam ainda uma demanda crescente por recursos ligados a sistemas inteligentes autônomos, o que torna a implementação de redes neurais artificiais em *hardware*, por exemplo, uma tendência para os próximos anos (OMONDI; RAJAPAKSE, 2006). Outras ações, como o desenvolvimento de sistemas orientados para a área educacional, que abrangem o uso das redes neurais artificiais também motivam diversos profissionais (PETERSON; NERY; ARAUJO, 2006).

Ainda no foco das aplicações, estudos importantes de redes neurais artificiais direcionadas à área da saúde estão auxiliando hoje pesquisadores, médicos, entre outros, no diagnóstico de doenças (DIAS; RADONSKY, 2003; FERNANDES et al., 2004).

Nos últimos anos surgiu uma diversidade de ferramentas computacionais voltadas para a implementação de agentes ou sistemas inteligentes. A própria criação e o aperfeiçoamento das linguagens de programação de alto nível propiciaram também o desenvolvimento de *softwares* e componentes destinados ao campo da inteligência artificial (MEDEIROS, 2006).

Ferramentas dedicadas à simulação computacional de modelos matemáticos, como o *software* Matlab®, oferecem enorme gama de possibilidades para a implementação e testes de redes neurais artificiais.

O conhecido *Toolbox* de redes neurais artificiais, existente no pacote Matlab® (MATHLAB, 1994), disponibiliza uma implementação de grande desempenho e baixo tempo de desenvolvimento. No entanto, além de a ferramenta estar voltada particularmente para cientistas e pesquisadores, o seu custo é relativamente alto. Cabe ressaltar ainda que outras ferramentas e linguagens de programação apresentam um custo importante, quando se trata de seu uso a partir de uma grande quantidade de licenças de *software*. Salienta-se também uma maior necessidade por recursos computacionais, no caso do *hardware*, para a instalação e uso do *software* Matlab® e seus componentes.

Nesse sentido, este trabalho apresenta uma proposta de implementação de redes neurais artificiais em ferramentas computacionais de custo reduzido, com amplo conhecimento da socieda-

de, em geral, e também da comunidade acadêmica, além da exigência de menos recursos de *hardware*, como o *software* Microsoft Excel®, que oferece um instrumento de programação conhecido como *Visual Basic for Applications* (VBA) (SYRSTAD; JELEN, 2004), no qual é possível a codificação de um algoritmo computacional, no caso específico de um algoritmo de uma rede neural artificial.

A programação da rede neural no ambiente (VBA) permite fácil interação das variáveis e dos dados envolvidos na lógica do sistema, com a estrutura de planilhas disponível no *software* Microsoft Excel®, como será demonstrado ao longo deste artigo. Naturalmente, o uso do referido ambiente sugere que o desenvolvedor tenha conhecimento das principais técnicas de programação existentes. De toda a forma, o seu aprendizado se justifica em função dos benefícios proporcionados pela implementação de um sistema inteligente dedicado.

É importante destacar ainda que este trabalho tem como finalidade também oferecer um ambiente mais amigável para o estudo das redes neurais artificiais por parte de estudantes de graduação e pós-graduação, bem como para profissionais que utilizam, com freqüência, o *software* Microsoft Excel® em diversas áreas, e para solucionar uma ampla variedade de problemas. Importantes ferramentas ou mesmo sistemas de informação podem ser desenvolvidos para auxiliar o usuário no processo de tomada de decisão.

De toda forma, convém mencionar que não é objetivo deste estudo explorar com profundidade todos os recursos oferecidos pelo *software* Microsoft Excel®, uma vez que o ambiente VBA e similares estão disponíveis em outras ferramentas, sendo abordadas apenas suas funcionalidades principais no que se refere à implementação de redes neurais artificiais.

## 2 A ferramenta Microsoft Excel® e o ambiente VBA (*Visual Basic for Applications*)

O ambiente VBA, presente no *software* Microsoft Excel®, será utilizado neste trabalho como suporte operacional para a implementação de redes neurais artificiais. A Figura 1 ilustra o ambiente disponível na ferramenta de software.

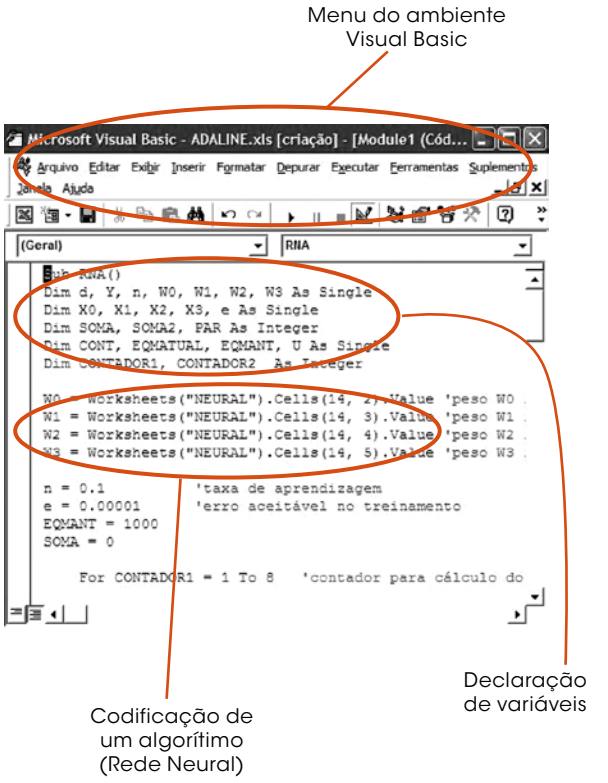


**Figura 1: O ambiente VBA disponível na ferramenta Excel®**
Fonte: O autor.

A Figura 1 mostra as variáveis declaradas para uma rede neural artificial no ambiente Excel®, tais como os pesos sinápticos (vetor W), a taxa de aprendizagem da rede (variável n) e o erro aceitável durante a fase de treinamento.

Na Figura 1, é possível notar ainda que a implementação do algoritmo da rede neural segue os mesmos procedimentos de uma programação de alto nível, orientada para uma linguagem de

programação padrão e/ou comercial. Nesse caso, a codificação é similar à linguagem de programação Microsoft Visual Basic®. Para iniciar a codificação da rede neural artificial, ou qualquer outra codificação por meio do ambiente *Visual Basic for Applications*, pode-se seguir as etapas ilustradas na Figura 2.

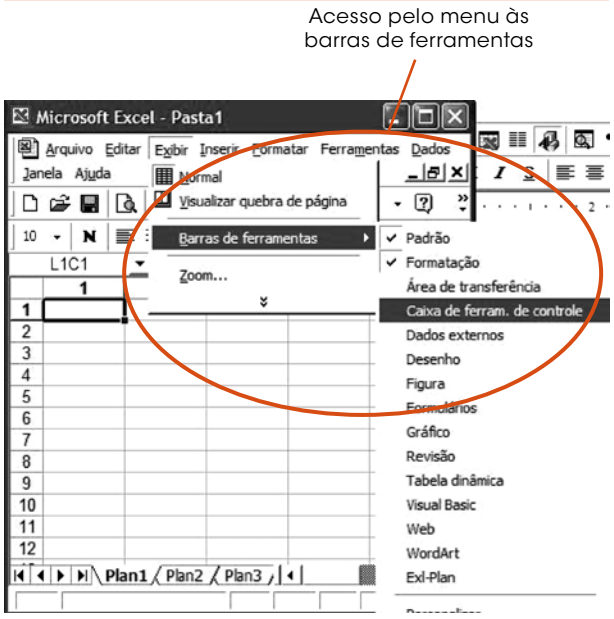


**Figura 2: Acesso às barras de ferramentas do *software***
Fonte: O autor.

Quando o usuário acessar o item "Caixa de ferram. de controle" em uma planilha do *software* Excel®, conforme indicado na Figura 2, aparecerá a caixa (Figura 3).

Na Figura 3, nota-se a existência de vários ícones destinados à edição de macros e programas no ambiente *Visual Basic for Applications*. Para iniciar a montagem de uma nova codificação, como o algoritmo de uma rede neural, objeto deste trabalho, pode-se inserir um novo botão em uma planilha, conforme ilustrado na Figura 4.

A partir de um duplo *click* no *CommandButton1* inserido na planilha, conforme indicado na Figura 4, pode-se editar o código do algoritmo desejado, desde que o ícone para edição

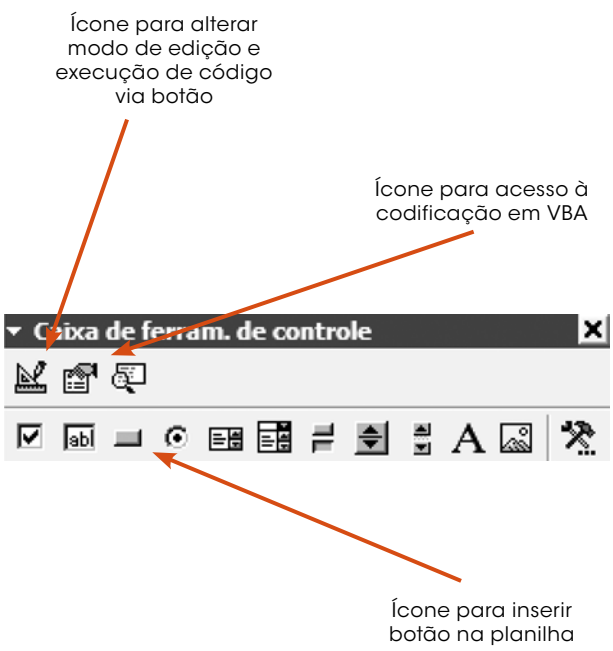


**Figura 3: Caixa de ferramentas de controle**
Fonte: O autor.

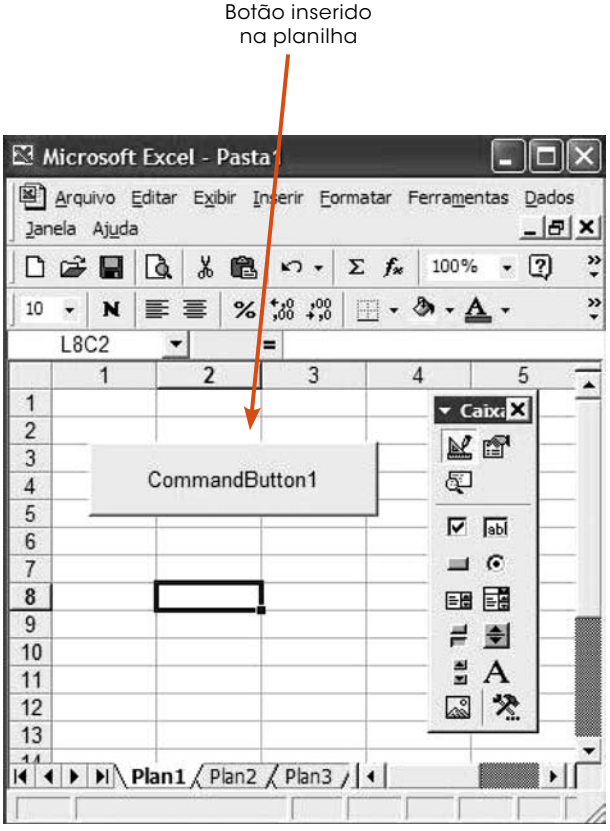


**Figura 4: Adicionando um botão na planilha do Excel®**
Fonte: O autor.

(pequeno esquadro) esteja no modo “estrutura”. Neste caso, surgirá o ambiente do *Visual Basic for Applications*, como ilustrado na Figura 1.

Com a inserção dos dados em uma planilha, é possível vinculá-los ao algoritmo codificado no ambiente *Visual Basic for Applications* (Figura 5). Na mesma figura, pode-se observar que a variável W0 (peso sináptico relacionado a uma das entradas de uma rede neural artificial), declarada no início do programa, recebe o valor inserido na linha 3 e coluna 2 da planilha. Neste caso, a variável receberá o valor igual a 15. A linha de código abaixo identifica a maneira pela qual as variáveis são referenciadas em uma codificação em VBA.

W0 = Worksheets(“Plan1”).Cells(3, 2).Value (1)

A expressão acima mostra que o nome da planilha, no caso Plan1, deve ser declarado para associar os valores do programa aos valores das células localizados na mesma planilha. No item *Cells*, localizado na expressão, os valores de linha e coluna devem ser declarados também para direcionar a célula desejada na referida planilha.

É possível implementar todo o algoritmo de uma rede neural no ambiente VBA e vinculá-lo aos dados cadastrados na planilha de origem. Como demonstrados no próximo item, os pesos sinápticos da rede neural, bem como outras informações, podem ser apresentados ao longo do treinamento da rede e durante sua execução após a fase de testes.

## 3 A implementação de uma rede neural no ambiente VBA

Como apresentado em (HAYKIN, 2001), o neurônio artificial é uma unidade de processamento de informações, sendo parte fundamental na operação de uma rede neural. Tal como mencionado anteriormente, uma rede neural artificial pode ser implementada hoje em inúmeras aplicações, a partir de um aprendizado obtido por meio de exemplos.

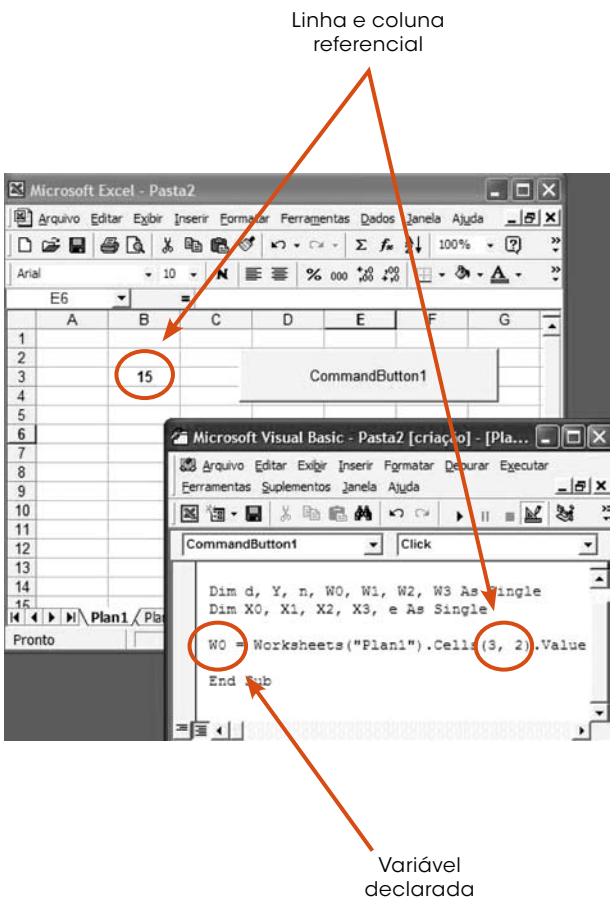


**Figura 5: Associando valores de uma planilha com a codificação em VBA®**
Fonte: O autor.

Neste sentido, uma rede neural é utilizada em aplicações as quais um processamento complexo se faz necessário. Para tanto, modelos matemáticos de redes neurais artificiais foram desenvolvidos e aperfeiçoados para implementações em ambientes computacionais. Os cientistas McCullock e Pitts propuseram na década de 1940 um modelo de neurônio artificial fundamentado no funcionamento de uma célula neural biológica.

Cabe destacar que o funcionamento de uma unidade de processamento, como aquela proposta por McCullock e Pitts, em 1943 (HAYKIN, 2001), pode ser descrita da seguinte maneira:

- os sinais são apresentados nas entradas (assim como na célula nervosa biológica);
- cada sinal é multiplicado por um número, ou peso, que indica sua influência na saída da unidade;
- é feita a soma ponderada dos sinais que produz um nível de atividade (assim como o limiar de disparo de uma célula nervosa biológica);
- se este nível de atividade exceder certo limite (*threshold*), a unidade produzirá determinada resposta de saída.

Neste item, demonstram-se a implementação de uma rede neural do tipo Perceptron Multicamadas no ambiente VBA, avaliada anteriormente, e os resultados do seu treinamento e testes após o seu aprendizado. Inicialmente, são apresentados a topologia da rede neural utilizada e sua fundamentação matemática para codificação e, em seguida, os resultados relativos ao seu aprendizado e à execução da rede. Sabe-se que o aprendizado de uma rede neural pode ser dividido em duas categorias: o aprendizado supervisionado e o não-supervisionado (AZEVEDO; BRASIL; OLIVEIRA, 2000).

A rede neural utilizada neste trabalho, cujo neurônio é modelado matematicamente na estrutura Perceptron, funciona a partir de um algoritmo de treinamento supervisionado. Neste caso, cada neurônio recebe o padrão de dados na sua entrada, realiza seu processamento e, em seguida, disponibiliza os valores associados na camada de saída da rede.

A rede formada por neurônios do tipo Perceptron é uma rede Feedforward (Multicamadas) constituída por uma camada de entrada, pelo menos por uma camada neural escondida (geralmente duas) e uma neural de saída.

A Figura 6 ilustra a estrutura da rede neural tipo Feedforward.

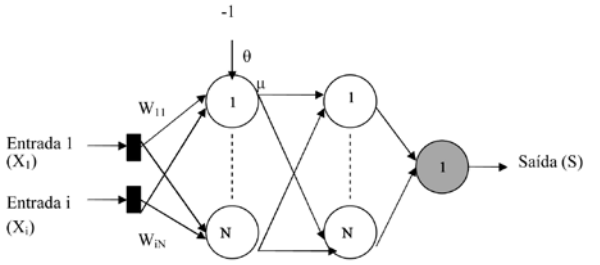


**Figura 6: Topologia da rede neural Feedforward**
Fonte: O autor.

Matematicamente, o neurônio 1, como ilustrado na Figura 6 e apresentado originalmente por McCullock e Pitts durante os seus trabalhos, pode ser representado por:

$$\mu = \sum_{i=1}^{N} W_i \cdot X_i - \theta \qquad (2)$$

$$\gamma = g(\mu) \qquad (3)$$

Os demais neurônios da rede seguem a mesma modelagem matemática.

A partir das equações 2 e 3, nota-se que a função de ativação g(μ) processa o conjunto de entradas recebidas por neurônio e o transforma em estado de ativação. A principal finalidade da função de ativação é limitar a saída do neurônio em intervalos conhecidos, pois, sem ela, a rede poderia assumir qualquer valor.

Neste trabalho, uma rede neural que utiliza atrasos de tempo para realizar processamento temporal, conhecida como TDNN (Time Delay Neural Network), foi implementada para a estimação de valores futuros a partir de um conjunto de dados que formavam a base de conhecimento

da rede neural. Mais particularmente, a referida rede foi escolhida com a finalidade de se prever o comportamento de um sistema. Como exemplo, neste trabalho foi utilizada uma função matemática para validar o comportamento da rede neural.

A rede TDNN é alimentada diante de múltiplas camadas, cujos neurônios ocultos e os de saída são replicados através do tempo (HAYKIN, 2001). É importante ressaltar que a rede neural artificial do tipo TDNN pode ser empregada em aplicações de previsões temporais voltadas, por exemplo, para uma análise de crédito ou para situações direcionadas a investimentos financeiros (aplicações em bolsa de valores).

A Figura 7 ilustra a estrutura final da rede neural implementada neste trabalho. Essa topologia foi desenvolvida com 8 neurônios na primeira camada escondida, e 12, na segunda. O número de neurônicos final para cada camada foi alcançado a partir do erro quadrático médio desejado, neste caso, para um valor próximo de $5x10^{-5}$. Assim, o número de neurônios foi alterado a partir de uma série de treinamentos, isto é, de modo empírico, até se alcançar o erro desejado.

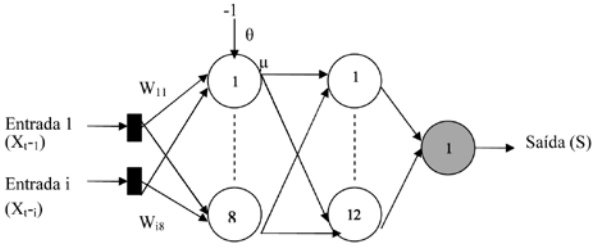


**Figura 7: Topologia da rede neural implementada**
Fonte: O autor.

Os dados para treinamento da rede neural foram gerados a partir da função matemática:

$$Y = 0.4 * |\text{sen}(x) + \cos(2x)| \quad (4)$$

A saída da rede neural, ilustrada na Figura 7, deve fornecer o valor desejado para a função matemática indicada na equação 4. A rede foi treinada com base nos primeiros 50 valores gerados na planilha, conforme ilustrado na Figura 8.

| | X1 | X2 | X3 | X4 | X5 | X6 | X7 | X8 | saida |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.007671 | 0.798414 | 0.092366 | 0.436561 | 0.019479 | 0.299827 | 0.25378 | 0.698704 | 0.433058 |
| 2 | 0.433058 | 0.007671 | 0.798414 | 0.092366 | 0.436561 | 0.019479 | 0.299827 | 0.25378 | 0.014035 |
| 3 | 0.014035 | 0.433058 | 0.007671 | 0.798414 | 0.092366 | 0.436561 | 0.019479 | 0.299827 | 0.354292 |
| 4 | 0.354292 | 0.014035 | 0.433058 | 0.007671 | 0.798414 | 0.092366 | 0.436561 | 0.019479 | 0.199455 |
| 5 | 0.199455 | 0.354292 | 0.014035 | 0.433058 | 0.007671 | 0.798414 | 0.092366 | 0.436561 | 0.736158 |
| 6 | 0.736158 | 0.199455 | 0.354292 | 0.014035 | 0.433058 | 0.007671 | 0.798414 | 0.092366 | 0.261331 |
| 7 | 0.261331 | 0.736158 | 0.199455 | 0.354292 | 0.014035 | 0.433058 | 0.007671 | 0.798414 | 0.420358 |
| 8 | 0.420358 | 0.261331 | 0.736158 | 0.199455 | 0.354292 | 0.014035 | 0.433058 | 0.007671 | 0.127124 |
| 9 | 0.127124 | 0.420358 | 0.261331 | 0.736158 | 0.199455 | 0.354292 | 0.014035 | 0.433058 | 0.218181 |
| 10 | 0.218181 | 0.127124 | 0.420358 | 0.261331 | 0.736158 | 0.199455 | 0.354292 | 0.014035 | 0.392431 |
| 11 | 0.392431 | 0.218181 | 0.127124 | 0.420358 | 0.261331 | 0.736158 | 0.199455 | 0.354292 | 0.539927 |
| 12 | 0.539927 | 0.392431 | 0.218181 | 0.127124 | 0.420358 | 0.261331 | 0.736158 | 0.199455 | 0.592477 |
| 13 | 0.592477 | 0.539927 | 0.392431 | 0.218181 | 0.127124 | 0.420358 | 0.261331 | 0.736158 | 0.329825 |
| 14 | 0.329825 | 0.592477 | 0.539927 | 0.392431 | 0.218181 | 0.127124 | 0.420358 | 0.261331 | 0.259945 |
| 15 | 0.259945 | 0.329825 | 0.592477 | 0.539927 | 0.392431 | 0.218181 | 0.127124 | 0.420358 | 0.08327 |
| . | . | . | . | . | . | . | . | . | . |
| . | . | . | . | . | . | . | . | . | . |
| . | . | . | . | . | . | . | . | . | . |
| 50 | 0.764361 | 0.184358 | 0.381281 | 0.022383 | 0.378545 | 0.105802 | 0.766344 | 0.216669 | 0.165445 |

**Figura 8: Dados utilizados para o treinamento da rede neural**
Fonte: O autor.

Uma parte da codificação da rede neural TDNN, implementada no ambiente VBA disponível na ferramenta Microsoft Excel®, está ilustrada na Figura 9.

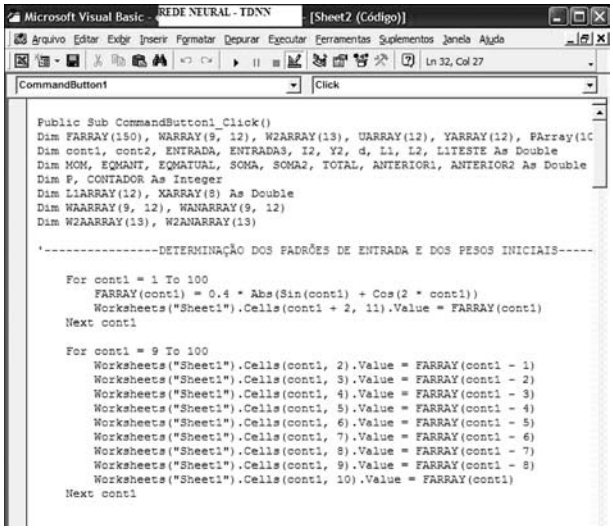


**Figura 9: Parte da codificação da rede neural implementada no ambiente VBA**
Fonte: O autor.

Nessa aplicação, a rede neural deve estimar os próximos valores a partir dos dados gerados

pela função matemática. A rede recebe em suas entradas os 8 valores anteriores gerados pela referida função e exibe em sua saída o valor imediatamente posterior.

Os dados poderiam sugerir o comportamento de um sistema qualquer, que poderia ser um processo industrial ou máquina, ou ainda, o comportamento de uma aplicação financeira, por exemplo, considerando as eventuais não-linearidades do problema abordado.

Durante a fase de treinamento da rede, os pesos sinápticos foram ajustados, para o devido aprendizado, em razão do erro desejado entre as suas entradas e respectivas saídas (treinamento supervisionado). Os pesos iniciais da rede neural, para a primeira camada escondida, foram inseridos em uma planilha da ferramenta Excel®, como ilustrado na Figura 10.

| | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | | | | | PESOS INICIAIS | | | | | | | |
| 3 | 1.1 | 0.8 | -0.9 | 0.11 | 0.23 | -0.622 | 0.3123 | 0.14 | -0.8723 | 0.7231 | -0.12378 | -0.40549 |
| 4 | 1.234 | 5.283 | 0.8232 | 0.87732 | 0.233 | -0.2345 | 0.6949131 | 0.152273968 | 0.060606 | 0.626708 | 0.694931 | 0.195842 |
| 5 | 1.1 | 0.8 | -0.9 | 0.11 | 0.23 | -0.622 | 0.3123 | 0.14 | -0.8723 | 0.7231 | -0.12378 | -0.40549 |
| 6 | 1.234 | 5.283 | 0.8232 | 0.87732 | 0.233 | -0.2345 | 0.6949131 | 0.152273968 | 0.060606 | 0.626708 | 0.694931 | 0.195842 |
| 7 | 1.1 | 0.8 | -0.9 | 0.11 | 0.23 | -0.622 | 0.3123 | 0.14 | -0.8723 | 0.7231 | -0.12378 | -0.40549 |
| 8 | 1.234 | 5.283 | 0.8232 | 0.87732 | 0.233 | -0.2345 | 0.6949131 | 0.152273968 | 0.060606 | 0.626708 | 0.694931 | 0.195842 |
| 9 | 1.1 | 0.8 | -0.9 | 5.143712397 | 0.052486367 | -0.622 | 0.3123 | 5.38607655 | -0.8723 | 0.7231 | -0.12378 | -0.40549 |
| 10 | 1.234 | 5.283 | 0.8232 | 0.87732 | 0.233 | -0.2345 | 0.6949131 | 0.870967544 | 0.060606 | 0.626708 | 0.694931 | 0.195842 |
| 11 | 0.9343 | 0.7643 | -0.4222 | -0.88423 | 0.8 | -0.9 | 0.11 | 5.17834289 | 0.060606 | 0.626708 | -0.8723 | 0.2233 |
| 12 | | | | | | | | | | | | |
| 13 | | | | | PESOS AJUSTADOS | | | | | | | |
| 14 | 0.5466655 | 0.818448 | -1.60263 | 0.84959115 | 0.342849494 | -9.6503809 | 1.0126 | -0.69677409 | -0.57296 | 1.593486 | 2.375963 | -0.22525 |
| 15 | 1.25943 | 5.308204 | 1.339627 | 1.631044983 | 0.239287669 | -1.5638217 | 0.864841 | 0.428803247 | 0.564603 | 0.863916 | 2.369241 | 0.256696 |
| 16 | 1.283423 | 0.812593 | -0.9573 | 1.06189357 | 0.088296701 | 0.0114992 | -0.89651 | 0.288343105 | -1.64321 | -0.61766 | -3.72007 | -0.70221 |
| 17 | 1.093366 | 5.281275 | 0.580566 | -0.68871355 | 0.796254001 | 0.496254 | -1.20476 | -0.19627737 | -0.07628 | 1.284833 | 2.431751 | 0.311396 |
| 18 | 0.678293 | 0.762213 | -1.59515 | -2.84555556 | 2.524722541 | 6.57372254 | 2.113923 | 2.168331653 | 1.524012 | 0.972288 | -8.63427 | -0.39192 |
| 19 | 1.766148 | 5.250508 | 1.101866 | 3.105084988 | 0.082684988 | -0.217315 | -1.91642 | -0.90611601 | -0.78612 | 0.86047 | 1.096819 | 0.276614 |
| 20 | 1.455759 | 0.794274 | 0.172266 | 5.066855045 | 0.168041754 | -2.0198286 | 0.171367 | 5.283438142 | -0.88996 | 1.01479 | 0.121867 | -0.35517 |
| 21 | 1.567191 | 5.271862 | 0.88266 | 1.516944538 | 0.304234528 | -9.1823499 | 0.948251 | 1.266639538 | 0.445253 | 0.682727 | -1.0443 | 0.315661 |
| 22 | 0.751581 | 0.798861 | 0.179239 | -1.23441224 | 0.768190236 | -2.4419622 | -0.14896 | 3.817607404 | 0.109562 | 0.537386 | 0.533958 | 0.157864 |

**Figura 10: Pesos sinápticos iniciais e ajustados na planilha do Excel®**
Fonte: O autor.

Os pesos iniciais e ajustados foram vinculados à codificação da rede no ambiente VBA, de acordo com a parte de código ilustrada na Figura 11. Maiores detalhes acerca do ajuste dos pesos sinápticos podem ser encontrados em (HAYKIN, 2001).

O algoritmo da rede neural é executado para minimizar o erro desejado entre suas entradas e respectivas saídas. No algoritmo da rede, o erro quadrático médio, calculado em cada iteração do programa, é apresentado na planilha até a sua convergência. Cabe salientar que o erro quadrático médio é um critério de parada para o algoritmo de treinamento de uma rede neural artificial. O referido erro calcula, durante o treinamento da rede, a diferença entre o valor gerado nas saídas de uma rede neural e o valor desejado. O número de épocas de treinamento, isto é, de iterações, é apresentado na planilha, em cada passo.

```
'Atribuição dos pesos iniciais

For cont2 = 2 To 13
    For cont1 = 1 To 9
    WARRAY(cont1, cont2) = Worksheets("NEURAL").Cells(cont1 + 2,
cont2).Value
    Next cont1
  Next cont2


'Pesos ajustados e vinculados à planilha denominada "NEURAL"

 For CONTADOR = 1 To 12
 Worksheets("NEURAL").Cells(12, CONTADOR).Value = WARRAY(1,
CONTADOR)
 For cont1 = 1 To 8
 Worksheets("NEURAL").Cells(cont1 + 12, CONTADOR).Value =
WARRAY(cont1 + 1, CONTADOR)
 Next cont1
 Worksheets("NEURAL").Cells(CONTADOR + 24, 5).Value =
W2ARRAY(CONTADOR + 1)
Next CONTADOR
```

**Figura 11: Pesos sinápticos iniciais e ajustados associados à codificação da rede**
Fonte: O autor.

Nessa aplicação, a rede neural foi treinada para aprender o comportamento de um sistema cujo sinal apresenta um comportamento orientado pela equação 4. A Figura 12 mostra o erro quadrático médio obtido durante a fase de aprendizado da rede.

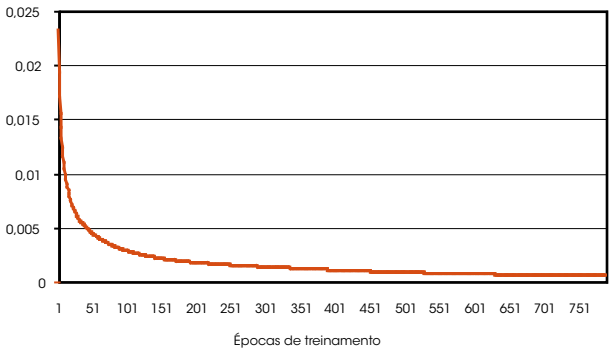


**Figura 12: Variação do erro quadrático médio durante o aprendizado da rede**
Fonte: O autor.

O gráfico ilustrado na Figura 12 foi montado a partir da barra de ferramentas do *software*

Microsoft Excel®, na opção "Inserir", e em seguida, "Gráfico". Para o aprendizado da referida rede, foram necessárias 786 épocas de treinamento, sendo de, aproximadamente, $5.053 \times 10^{-5}$ o erro quadrático médio final.

Os resultados obtidos pela rede neural, após o seu treinamento, foram comparados com os produzidos pela função matemática (4), como mostra a Figura 13. Neste caso, a rede neural estimou os 50 valores posteriores aos dados utilizados na fase de aprendizado da rede e gerados pela função matemática.

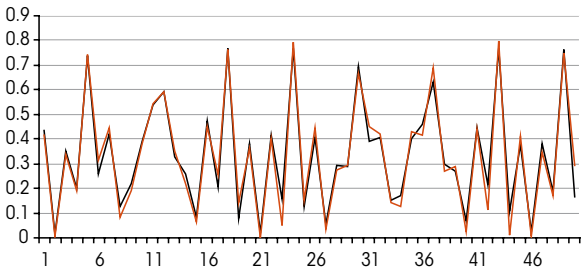


**Figura 13: Comparação dos dados gerados pela rede neural (azul) com aqueles produzidos pela função matemática**

Fonte: O autor.

Na Figura 13, é possível observar a boa aproximação dos dados gerados pela rede neural com aqueles obtidos pela função matemática.

## 4 Considerações finais

Apresentou-se e abordou-se a implementação de redes neurais artificiais em uma ferramenta comercial e amplamente utilizada: o Microsoft Excel®. Conforme discutido, ao longo do artigo, o *software* oferece um ambiente de programação conhecido como *Visual Basic for Applications (VBA)*, sendo possível o desenvolvimento de uma diversidade de algoritmos e soluções em diversas áreas. No tocante à plataforma de *hardware* escolhida, para o presente trabalho optou-se pelo *software* comercial Microsoft Excel® instalado em um computador pessoal (PC), facilmente encontrado e amplamente utilizado nos dias de hoje.

Cabe ressaltar que esta aplicação foi implementada com a finalidade de oferecer novos recursos para os usuários e profissionais no processo de tomada de decisão, em um ambiente corporativo, acadêmico ou científico. Acredita-se que o objetivo inicial tenha sido alcançado, uma vez que uma nova ferramenta para o estudo e testes de redes neurais artificiais pode ser disponibilizada, de forma eficiente e com o custo reduzido.

Finalmente, é importante salientar que novas aplicações utilizando o mesmo tipo de ferramenta, além da implementação de outras técnicas de inteligência artificial, como a lógica *fuzzy* ou os algoritmos genéticos, podem ser desenvolvidas sem grandes dificuldades pelo desenvolvedor que conheça algoritmos inteligentes. Convém mencionar, ainda, que outras ferramentas comerciais podem ser utilizadas, desde que ofereçam ambientes de programação como o VBA ou similares, adequados à implementação de algoritmos computacionais.

## The implementation and the study of artificial neural networks in commercial software tools

In this paper, it is presented an approach about the implementation of artificial neural networks in commercial software tools, in which is possible the study and evaluate of intelligent systems for academic purposes or for professional users of many knowledge areas, in order to help them at the decision process. The methodology of neural network implementation in a software tool like Microsoft Excel® and their results are presented in this work.

**Key words:** Artificial intelligence. Information systems. Neural networks.

## Referências


AQUINO, R. R. B.; CARVALHO, M. A.; SOUZA, B. A. Redes Neurais de Hopfield como ferramenta de otimização aplicada ao despacho hidrotérmico. In: *of the* IV BRAZILIAN CONFERENCE ON NEURAL NETWORKS, 4., 1999, São José dos Campos. *Proceedings*... São José dos Campos, SP, Brazil. 1999. p. 170-175.

AZEVEDO, F. M.; BRASIL, L. M.; OLIVEIRA, R. C. L. *Redes neurais com aplicações em controle e em sistemas especialistas*. 1 ed. Florianópolis: Visual Books, 2000.

DIAS, C.G; RADONSKY, V. Desenvolvimento de um sistema de auxílio ao diagnóstico em pediatria com o uso de redes neurais artificiais. *Revista Exacta*, São Paulo, v.1, p.89-95, 2003.

FERNANDES, A. M. R. F. et al. *Inteligência artificial aplicada à saúde*. 1 ed. Florianópolis: Visual Books, 2004.

HAYKIN, S. *Redes neurais*: princípios e prática. 2 ed. Porto Alegre: Bookman, 2001.

MATHLAB. *Neural Network Toolbox*, 1994.

MEDEIROS, L. F. *Redes neurais em Delphi*. 2 ed. Florianópolis:Visual Books, 2006.

OMONDI, A. R.; RAJAPAKSE, J. C. *FPGA implementations of neural networks*. E. Springer, Netherlands, 2006

PALADE, V.; BOCANIALA, C. D; JAIN, L. C. *Computational intelligence in fault diagnosis*. Ed., United States, Springer, 2006.

PETERSON, A. B.; NERY, E. P.; ARAÚJO, S. A. Software para auxílio à pré-alfabetização infantil baseado em reconhecimento inteligente de caracteres manuscritos. *Revista Exacta*, São Paulo, v. 4, n.1, p.87-93, 2006.

PLANTULLO, V.L. *Teoria geral da administração:* de Taylor às redes neurais. 2. ed. Rio de Janeiro: FGV Editora, 2002.

RUSSEL, S.; NORVIG, P. *Artificial intelligence*: a modern approach. 1 ed. Nova Jersey: Prentice Hall, 1995.

SYRSTAD, T.; JELEN, B. *Macros e VBA para Microsoft Excel intermediário/avançado*. 2. ed. Campus, 2004.



Recebido em 25 fev. 2008 / aprovado em 12 maio 2008

**Para referenciar este texto**

DIAS, C. G. A implementação e o estudo de redes neurais artificiais em ferramentas de *software* comerciais. *Exacta*, São Paulo, v. 6, n. 1, p. 119-128, jan./jun. 2008.