Seleção de características de dados utilizando Redes Neurais Artificiais
Alternative title: Data features selection using Artificial Neural Network

Álvaro H. Nogueira de Lima Wilian Soares Lacerda Heitor Scalco Neto
Laboratório de Sistemas Inteligentes e Embarcados
Departamento de Ciência da Computação – Universidade Federal de Lavras
Caixa postal 3037 – Campus Universitário da UFLA
Lavras - MG - CEP 37.200-000
nogueira.alvaro@gmail.com lacerda@dcc.ufla.br heitorscalco@hotmail.com

RESUMO

As Redes Neurais Artificiais (RNAs) tem se difundindo ao longo dos anos e sua utilização vem crescendo devido aos bons resultados encontrados na solução de diversos problemas do mundo real. Porém, a presença de variáveis de entrada sem importância ou redundantes que nada acrescentam ao processo de aprendizagem das RNAs tornam o seu treinamento mais difícil e demorado. Os métodos de seleção de características têm por objetivo determinar quais variáveis (características) da entrada são mais relevantes para a determinação da saída ou resposta da RNA, e assim auxiliar na redução do número de entradas. Neste trabalho foram implementados e avaliados quatro métodos de seleção de características baseados em RNAs: método de Garson; Perturb; PaD; e Análise de Sensibilidade. Todos os métodos foram comparados com os resultados obtidos pelo método estatístico clássico de Correlação Linear. Os dados de três problemas reconhecidos na área (Íris, Desempenho da CPU, Resistência do concreto) foram utilizados para o treinamento de RNAs que, após treinadas utilizando o algoritmo *Error Backpropagation*, os métodos de seleção de características foram executados obtendo-se a importância de cada entrada. Para os dados do problema Íris, todos métodos apresentaram resultados semelhantes. Para os problemas Desempenho da CPU e Resistência do Concreto, o método Perturb apresentou os piores resultados, o método de Garson obteve um resultado satisfatório, e os métodos PaD e Análise de Sensibilidade apresentaram melhores resultados se destacando em relação aos demais.

Palavras-Chave

Redes Neurais Artificiais; seleção de características; Correlação Linear.

ABSTRACT

Artificial Neural Networks (ANNs) has been spreading over the years and they have been growing due to the good results found in solving various real world problems. However, the presence of unimportant or redundant input variables that add nothing to the learning process of ANNs makes their training more difficult and time-consuming. The characteristic selection methods are aimed at determining which input variables are most relevant for the determination of ANN output or response, thus helping to reduce the number of inputs. In this work four methods of selection of characteristics based on ANNs were implemented and evaluated: Garson method; Perturb; PaD; and Sensitivity Analysis. All methods were compared with the results obtained by the classical statistical method of Linear Correlation. The data of three recognized problems in the area (Iris, CPU Performance, Concrete Strength) were used for the training of ANNs that, after being trained using the Error Backpropagation algorithm, the characteristic selection methods were executed, obtaining the importance of each input. For the Iris problem, all methods presented similar results. For the CPU Performance and Concrete Resistance problems, the Perturb method presented the worst results, the Garson method obtained a satisfactory result, and the PaD and Sensitivity Analysis methods presented better results and they stood out in relation to the others.

SBSI 2017, June 5th–8th, 2017, Lavras, Minas Gerais, Brazil.

CCS Concepts

• Computing methodologies→Canonical correlation analysis.

Keywords

Artificial Neural Networks; features selection; Linear Correlation.

1. INTRODUÇÃO

Redes Neurais Artificiais (RNAs) fazem parte de uma classe da área de Inteligência Computacional cujo funcionamento tenta reproduzir o funcionamento de uma rede de neurônios biológicos (cérebro) de forma simplificada [11][13][14][16]. Ao longo dos anos sua utilização vem sendo difundida devido aos bons resultados encontrados na solução de diversos problemas do mundo real [3][6] e também devido à evolução do hardware e software que tornaram sua implementação mais intuitiva e rápida.

Diversos são os tipos de problemas nos quais as RNAs podem ser aplicadas [19], dentre eles podem-se destacar principalmente os problemas de classificação, agrupamento, reconhecimento de

padrões e aproximação de funções. O elevado grau de complexidade presente nas RNAs torna sua computação difícil [15][20], e em muitos casos, seu tempo de treinamento e custo computacional se torna muito elevado. Um dos fatores que prejudicam o desempenho da rede neural é a presença de variáveis (características ou atributos) de entrada redundantes que nada acrescenta ao seu processo de aprendizagem, tornando assim o treinamento mais difícil e demorado.

Visando contornar esses problemas, muitas técnicas e métodos de seleção de características de dados utilizando as próprias RNAs vêm sendo propostos ao longo do tempo. Estes métodos de seleção de características têm por objetivo determinar quais variáveis (características) da entrada são mais relevantes ou que mais contribuem para a determinação da saída ou resposta da rede. Através da seleção de características é possível eliminar as variáveis menos importantes e reduzir a dimensão dos dados de entrada fazendo com que o treinamento da rede neural seja mais rápido e eficiente.

O objetivo deste trabalho foi analisar algumas técnicas de seleção de características utilizando RNAs encontradas na literatura comparando os resultados obtidos com a técnica de Correlação Linear, amplamente utilizada em estatística. Assim, pretendeu-se determinar qual método de seleção de características apresenta melhores resultados para alguns exemplos de problemas.
O presente artigo é organizado da seguinte maneira: na seção 2 é apresentado um revisão geral sobre RNAs e seu treinamento; na seção 3 aão apresentados os quatro algoritmos de seleção de características de dados abordados neste trabalho, além da Correlação Linear; na seção 4 são apresentados os detalhes da metodologia adotada para implementação dos testes realizados; na seção 5 são apresentados e discutidos os resultados obtidos com cada conjunto de dados dos problemas escolhidos para teste; por fim, na seção 6, são apresentadas as conclusões deste trabalho.

2. REDES NEURAIS ARTIFICIAIS

RNAs (Redes Neurais Artificiais) podem ser consideradas como uma técnica utilizada em resolução de diversos tipos de problemas característicos da Inteligência Computacional dentre eles: previsão, classificação, reconhecimento de padrões e agrupamento. As RNAs são inspiradas no funcionamento do cérebro humano as quais, segundo Barreto [1], são construídas utilizando um sistema que modela os circuitos cerebrais esperando-se ver um comportamento inteligente emergindo, aprendendo novas tarefas, errando, fazendo generalizações e descobertas, e frequentemente ultrapassando seu professor. Segundo Haykin [10] uma rede neural artificial pode ser definida como: “um processador maciçamente paralelamente distribuído constituído de unidades de processamento simples, que têm a propensão natural para armazenar conhecimento experimental”.

A maneira como os neurônios de uma rede neural estão organizados e conectados uns aos outros neurônios é o que define a arquitetura da Rede Neural. Em geral, as RNAs estão enquadradas em três diferentes tipos de arquiteturas: feedforward de camada única, feedforward múltiplas camadas e redes recorrentes. A seguir são detalhados os dois primeiros tipos de arquiteturas; as redes recorrentes não serão apresentadas por não serem utilizadas neste trabalho.

2.1 Redes Feedforward de camada única

Esta é a forma mais simples de se implementar uma rede em camadas, na qual a rede possui uma camada de entrada de nós que se conectam aos neurônios (nós computacionais) da camada de saída. Os sinais são transmitidos somente na direção da entrada para a saída, daí o nome feedforward (“alimentadas adiante”) [10].

Apesar de possuir “duas” camadas, este tipo de rede neural é denominado de camada única devido ao fato da mesma possuir apenas uma camada de nós computacionais, desconsiderando-se assim a camada de entrada que somente propaga o sinal da entrada não realizando nenhum tipo de computação. A Figura 1 apresenta um exemplo de rede feedforward de camada única.

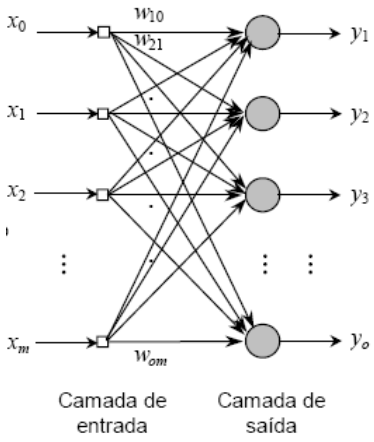

Figura 1. Rede feedforward de uma única camada [10].

2.2 Redes Feedforward de múltiplas camadas

Redes feedforward de múltiplas camadas (Figura 2), como o próprio nome sugere, possuem além da camada de entrada e saída (nós computacionais) pelo menos uma camada intermediária (ou escondida) em sua arquitetura, sendo que a camada de entrada propaga seu sinal à camada escondida e a saída da camada escondida é propagada para frente até a camada de saída passando pelas demais camadas escondidas [10].

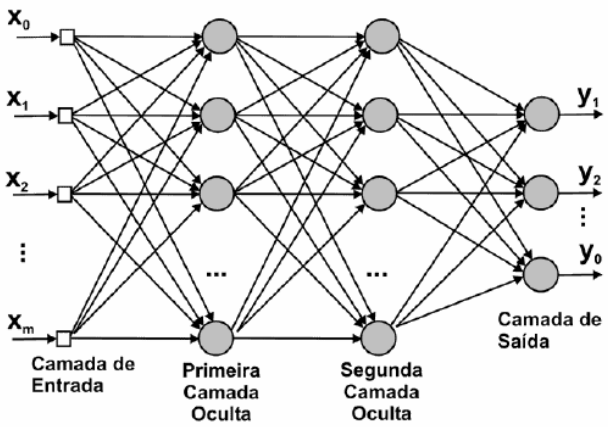

Figura 2. Rede feedforward de múltiplas camadas [10].

2.3 Treinamento das RNAs

A habilidade de aprender a partir de seu ambiente e de melhorar seu desempenho através da aprendizagem é a propriedade de maior importância para uma rede neural. Uma rede neural aprende através de um processo interativo chamado regra ou algoritmo de aprendizagem que consiste em correções ou ajustes de seus pesos sinápticos e bias [10]. O processo de aprendizagem é composto dos seguintes eventos:

1. A rede neural é estimulada por um ambiente;
2. A rede neural sofre modificações nos seus parâmetros como resultado dessa estimulação;
3. A rede neural responde de uma maneira nova ao ambiente.

Diversos métodos para treinamento de redes foram desenvolvidos, podendo ser separados em dois paradigmas principais: aprendizado supervisionado (Figura 3) e aprendizado não-supervisionado. Outros dois paradigmas bastante conhecidos são os de aprendizado por reforço (que é um caso particular de aprendizado supervisionado) e aprendizado por competição (que é um caso particular de aprendizado não-supervisionado).

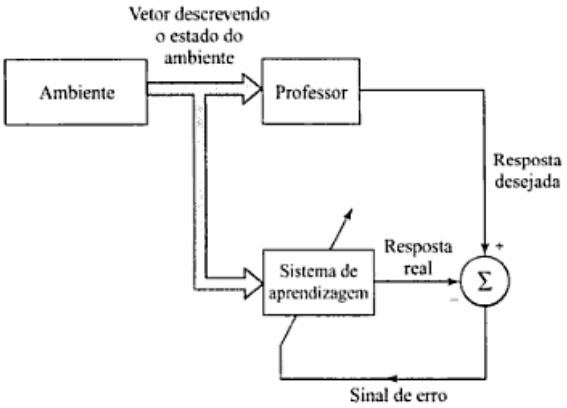

Figura 3. Mecanismo de aprendizado supervisionado [10].

A desvantagem da utilização do aprendizado supervisionado é que a rede não conseguirá aprender novas estratégias para situações não cobertas pelos exemplos do treinamento da rede na ausência do professor [2].

O aprendizado supervisionado pode ocorrer de duas maneiras: *off-line* e *on-line*. No treinamento *off-line,* uma vez obtida uma solução para a rede, os dados do conjunto de treinamento não mudam. Caso novos dados sejam adicionados ao conjunto de treinamento, um novo treinamento envolvendo também os dados anteriores deve ser realizado para se evitar interferência no treinamento anterior. Por outro lado, no aprendizado *on-line,* o conjunto de dados muda continuamente, e a rede deve estar em contínuo processo de adaptação [2]. Os exemplos mais conhecidos de algoritmos para aprendizado supervisionado são a regra delta e a sua generalização para redes de múltiplas camadas, o algoritmo *Error Backpropagation*.

2.3.1 Error Backpropagation

O *Error Backpropagation* é o algoritmo mais conhecido e mais utilizado para o treinamento de redes multi-camadas e pode ser definido segundo Braga [2] como: "um algoritmo supervisionado que utiliza pares (entrada, saída desejada) para, por meio de um mecanismo de correção de erros, ajustar os pesos da rede". O treinamento utilizando o *Error Backpropagation* ocorre em duas fases em que cada fase percorre a rede em uma única direção. Estas duas fases são chamadas de fase *forward e* fase *backward*. Durante a fase *forward* é feito o cálculo das saídas e seus respectivos erros. A fase *backward* utiliza o erro calculado durante a fase *forward* para fazer a atualização dos pesos de suas conexões.

2.3.2 Dificuldades do treinamento

Aprendizagem pode se tornar cada vez mais difícil à medida que o número de exemplos necessários para se aprender um certo conceito cresce exponencialmente com o número de características (variáveis de entrada) de cada exemplo de dados. Este problema é conhecido como maldição da dimensionalidade (*curse of dimensionalty*) [18].

Outro grande problema enfrentado no treinamento das RNAs é o *overfitting*. Nesta situação, a rede neural perde sua capacidade de generalização e passa a memorizar os exemplos de treinamento. Este problema ocorre geralmente devido ao elevado número de neurônios na camada escondida e também pelo excesso de treinamento. No entanto, se o número de neurônios na camada escondida for muito baixo, ocorrerá o *underfiting,* situação na qual a rede não converge durante seu treinamento.

3. SELEÇÃO DE CARACTERÍSTICAS

Seleção de características, também conhecida como seleção de variáveis ou seleção de atributos, é uma técnica que consiste na seleção de um subconjunto das características mais relevantes para a construção robusta de modelos de aprendizagem. Esta seção descreve alguns dos vários métodos de seleção de características utilizando RNAs encontrados na literatura.

3.1 Algoritmo de Garson

O algoritmo original proposto por Garson [8] envolve essencialmente o particionamento dos pesos das conexões entre a camada escondida e a de saída de cada neurônio intermediário em componentes associados com cada neurônio de entrada. Para a obtenção da importância relativa de cada variável o algoritmo executa os seguintes passos:

1 - Para cada neurônio intermediário $\boldsymbol{i}$, o valor absoluto do peso ($\boldsymbol{w_{io}}$) da conexão entre este neurônio e um de saída (neurônio **o**) é multiplicado pelo valor absoluto do peso da conexão entre o mesmo neurônio escondido e um neurônio de entrada. Este cálculo deve ser feito para todos os $\boldsymbol{j}$–ésimos neurônios da camada de entrada. Então o produto P_{ij} é obtido através da Equação 1:

$$\boldsymbol{P_{ij} = w_{ij}\, x\, w_{io}} \tag{1}$$

2 - Para cada neurônio escondido, divide-se $\boldsymbol{P_{ij}}$ pela soma de todos os $\boldsymbol{P_{ij}}$ para cada neurônio de entrada, obtendo $\boldsymbol{Q_{ij}}$. Assim:

$$Q_{ij} = P_{ij} / \sum_{j=1}^{n} P_{ij} \tag{2}$$

3 - Para cada neurônio de entrada, os valores de $\boldsymbol{Q_{ij}}$ são somados obtendo-se $\boldsymbol{S_j}$:

$$S_j = \sum_{i=1}^{n} Q_{ij} \tag{3}$$

4 - Dividindo-se cada valor de S_j pela soma de todos os valores de S_j, obtemos a importância relativa R para cada variável j:

$$R_j = \left(\frac{S_j}{\sum_{j=1}^{n} S_j} \right) \times 100 \quad (4)$$

O algoritmo de Garson utiliza valores absolutos dos pesos das conexões para o cálculo da contribuição da variável, não permitindo uma análise da direção das modificações ocorridas na variável de saída quando ocorre alteração nas variáveis de entrada [17].

3.2 Método Perturb

Este método tem como objetivo avaliar o efeito de pequenas mudanças em cada variável de entrada na saída da rede neural. O algoritmo ajusta os valores de entrada de uma variável, enquanto mantem todas as outras inalteradas. As respostas da variável de saída em relação a cada mudança na variável de entrada são armazenadas.

A variável de entrada cujas mudanças afetam mais a saída, será a que exerce maior influência relativa no modelo. Na verdade, é esperado que o erro médio quadrático (EMQ) da saída da rede neural aumente à medida que uma maior quantidade de ruído é adicionado à variável de entrada selecionada [21].

As mudanças nas variáveis, Equação 5, ocorrem geralmente adicionando à variável selecionada de 10% a 50% (valores mais utilizados) de seu valor [9].

$$x_i = x_i + \delta \quad (5)$$

em que x_i é a variável de entrada selecionada e δ é a mudança (10% a 50%) a ser adicionado a x_i.

3.3 Análise de Sensibilidade

Este método baseia-se nas derivadas parciais da variável de saída com relação aos pesos nas conexões, isto é, o método utiliza os resultados obtidos das derivadas parciais (sensibilidade) durante o treinamento da rede com o algoritmo *Error Backpropagation*.

A sensibilidade das variáveis de entrada é calculada através da retro-propagação do erro pelo algoritmo *Error Backpropagation*. Primeiro é feito o cálculo do erro para cada neurônio da camada de saída por meio do valor calculado (o_k) e a resposta desejada (d_k) (Equação 6).

$$e_k = d_k - oi_k \quad (6)$$

A sensibilidade de cada variável é calculada para cada exemplo T (T varia de 1 até n em que n é o número total de exemplos de treinamento) de acordo com a seguinte equação:

$$Sen_{jC} = \sum_{k=1}^{N_{hid}} w_{kj} \cdot f'(net_k) \times \sum_{i=1}^{N_{out}} w_{ik} \cdot f'(net_i) \cdot e_i \quad (7)$$

em que: Sen_{jC} é a sensibilidade para cada variável j de entrada em relação a saída para um exemplo de treinamento C; w_{kj} são os pesos sinápticos das conexões entre os neurônios escondidos e a camada de entrada; $f'(net_k)$ é a derivada da função de ativação dos neurônios da camada escondida; N_{hid} é o número total de neurônios na camada escondida; w_{ik} são os pesos sinápticos das conexões entre os neurônios da camada escondida e os da camada de saída; e $f'(net_i)$ é a derivada da função de ativação dos neurônios da camada de saída.

A partir da Sen_{jC} a contribuição percentual de cada variável j de entrada com relação a variável de saída é calculada por:

$$cont_j = \frac{\sum_{C=1}^{N} Sen_{jC}^2}{\sum_{j=1}^{N_{imp}} \sum_{C=1}^{N} Sen_{jC}^2} \quad (8)$$

em que: N representa o número total de exemplos de treinamento e N_{imp} é o número de entradas (neurônios na camada de entrada).

De acordo com Valença [17], os resultados de sensibilidade e de contribuição obtidos por este método permitem que sejam realizadas duas análises: uma análise da variação da saída para pequenas mudanças nas variáveis de entrada; e uma classificação da importância de cada uma das variáveis de entrada com relação a variável de saída da rede.

3.4 Método PaD

Este método, assim como o anterior, baseia-se em derivadas parciais. Porém, este método calcula a derivada parcial da saída da rede em relação à entrada. O perfil das variações da saída para pequenas alterações de uma variável de entrada é obtido através do cálculo da derivada parcial da saída da RNA em relação à entrada [5].

Para uma rede com n_i entradas, uma camada escondida com n_h neurônios e uma saída, a derivada parcial da saída y_j em relação à entrada x_j ($j = 1,...,N$, em que N é o total de amostras) é calculada de acordo com a equação a seguir:

$$d_{ji} = f'(net_j) \sum_{h=1}^{n_h} w_{ho} f'(net_h) w_{ih} \quad (9)$$

em que: $f'(net_j)$ e $f'(net_h)$ são as derivadas das funções de ativação dos neurônios da camada escondida e da camada de saída, respectivamente; w_{ho} e w_{ih} são os pesos entre o neurônio de saída e o h-ésimo neurônio escondido, e entre o i-ésimo neurônio de entrada e o h-ésimo neurônio escondido.

A contribuição relativa de uma entrada para a saída da RNA é calculada através da soma dos quadrados das derivadas parciais obtida pela seguinte equação:

$$SSD_i = \sum_{i=1}^{N} (d_{ji})^2 \quad (10)$$

Um valor SSD (*Sum of Square Derivatives*) é obtido para cada variável i de entrada, e a que apresentar o maior valor consequentemente será a que mais exerce influência na determinação da saída.

4. METODOLOGIA

Para a implementação das RNAs utilizadas neste trabalho foi utilizado o ambiente de programação Scilab 5.3.3. O Scilab é um software livre distribuído sobre CeCILL license (compatível com a GPL) para computação numérica e conta com liguagem de

programação própria de alto nível proporcionando um ambiente de computação poderosa para aplicações de engenharia científica. O Scilab por ser um software livre ainda conta com uma série de módulos desenvolvidos por usuários e outras empresas, aumentando ainda mais sua gama de aplicações. Neste trabalho em específico será utilizado o módulo *ANNToolbox* versão 0.5.2, desenvolvido por Hristev [12], que contém todas as funções necessárias para a implementação e treinamento de RNAs.

4.1 Base de dados

Esta seção descreve os problemas e os dados que foram utilizados nos treinamentos das RNAs implementadas, e na extração da importância de cada atributo dos dados. Os dados utilizados para o treinamento das RNAs foram obtidos do repositório da UCI *Machine Learning* [7].

4.1.1 Desempenho da CPU

O objetivo deste problema é determinar o desempenho relativo de uma CPU tendo como base algumas variáveis de entrada:

1. nome do fabricante: adviser, amdahl, apollo, basf, bti, burroughs, c.r.d, cambex, cdc, dec, dg, formation, four-phase, gould, honeywell, hp, ibm, ipl, magnuson, microdata, nas, ncr, nixdorf, perkin-elmer, prime, siemens, sperry, sratus, wang
2. nome do modelo: muitos símbolos
3. MYCT: tempo do ciclo de máquina em nanosegundos
4. MMIN: memória principal mínima em kilobytes
5. MMAX: memória principal máxima em kilobytes
6. CACH: memória cache in kilobytes
7. CHMIN: número mínimo de canais
8. CHMAX: número máximo de canais
9. PRP: desempenho relativo divulgado

As variáveis 1 e 2 por apenas conter nomes de marcas e modelos foram descartadas no treinamento. A base de dados para este problema conta com 209 exemplos ao todo e pode ser encontrada no repositório da UCI *Machine Learning* [7].

4.1.2 Íris

Originalmente o objetivo deste problema é fazer a classificação de algumas espécies da planta Íris (Iris-setosa, Iris-versicolor e Iris-virginica) levando em consideração os seguintes atributos:

1. Comprimento das sépalas (cm)
2. Largura das sépalas (cm)
3. Comprimento das pétalas (cm)
4. Largura das pétalas (cm)

Como alguns métodos de seleção de características restringem a rede neural a possuir apenas uma saída (Ex. Garson), o problema foi modificado para classificar apenas duas espécies (Iris-setosa, Iris-versicolor). A base de dados para este problema conta com 100 exemplos ao todo sendo que 50% são da espécie Iris-setosa e 50% da espécie Iris-versicolor. Essa base de dados pode ser encontrada no repositório da UCI *Machine Learning* [7].

4.1.3 Resistência do concreto.

A resistência do concreto é uma função altamente não linear da idade e componentes. Então, o objetivo deste problema é estimar a resistência do concreto à compressão levando em conta a idade e os componentes especificados abaixo:

1. Cimento [kg /m^3]
2. Resíduos do alto forno [kg/m^3]
3. Resíduo da combustão [kg/m^3]
4. Água [kg/m^3]
5. Superplastificante [kg/m^3]
6. Agregado graúdo [kg/m^3] (todo o agregado que fica retido na peneira de número 4, malha quadrada com 4,8 mm de lado)
7. Agregado miúdo [kg/m^3] (todo agregado que consegue passar por esta peneira)
8. Idade de 1 a 365 dias

A base de dados para este problema conta com 1030 exemplos ao todo e pode ser encontrada no repositório da UCI *Machine Learning* [7].

4.2 Pré-processamento dos dados

Normalmente as variáveis dos dados de entrada estão em intervalos de variação bastante distintos. Isto faz com que o treinamento e aprendizagem da RNA fiquem prejudicados, pois a rede neural pode interpretar valores mais altos como de maior importância e valores menores como menos importantes.

Para contornar este problema é utilizada a técnica de normalização dos dados. Isto torna todos os atributos com valores no intervalo da função de ativação dos neurônios. No caso deste trabalho, o Scilab adota como padrão a função de ativação sigmoidal que possui o intervalo variando de 0 a 1 [4]. Assim, todas as variáveis dos dados de entrada da rede foram normalizados de acordo com a Equação 11.

$$y = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}} \tag{11}$$

em que $\boldsymbol{y}$ é o valor normalizado; $\boldsymbol{x_i}$ é o valor original; $\boldsymbol{x_{min}}$ e $\boldsymbol{x_{max}}$ são, respectivamente, o valor mínimo e o valor máximo do conjunto de dados para determinado atributo.

Além do problema de grandes variações no intervalo dos dados, alguns atributos encontram-se com valores nominais o que torna seu processamento pela rede impossível de ser executado. A técnica mais utilizada para resolver este problema é a binarização, que consiste em atribuir números binários aos dados com valores nominais.

Os dados obtidos do repositório da UCI Machine Learning [7] foram separados aleatoriamente em duas categorias: dados de treinamento (80%), que serão utilizados para o treinamento da rede; e dados de teste (20%), que serão utilizados para verificar seu desempenho sob condições reais de utilização.

4.3 Configuração da rede

As RNAs utilizadas neste trabalho são do tipo feedforward (camada única e múltiplas camadas) com função de ativação sigmoidal e foram treinadas utilizando o algoritmo *Error Backpropagation* com *momentum* [10]. Para otimizar o treinamento alguns parâmetros foram inicializados manualmente com valores próximos de zero como no caso da taxa de aprendizagem e *momentum*. Os demais parâmetros foram inicializados automaticamente pelas funções implementadas pelo *toolbox* de RNAS. Para as redes implementadas, os valores de 0.1548 para a taxa de aprendizado e 0.238 para momentum foram os que apresentaram melhores resultados no treinamento.

Para simplificar a implementação dos métodos de seleção de características foi utilizada apenas uma camada escondida. A Equação 12 serve como um parâmetro para a definição do número total de neurônios na camada escondida.

$$N_{hid} = \frac{N_{out} + N_{inp}}{2} \quad (12)$$

em que N_{hid} é o número de neurônios na camada escondida, N_{out} é o número de neurônios na camada de saída, N_{inp} é o número de entradas da rede neural. Como citado anteriormente, essa equação serve apenas como um parâmetro para a definição do número de neurônios na camada escondida. Este número pode ser ajustado para mais ou para menos dependendo dos resultados e do problema em questão.

Com o intuito de se obter resultados mais consistentes, para cada problema foram treinadas dez RNAs, executando-se em seguida os algoritmos de seleção de características em cada uma delas. Uma vez executados os algoritmos, foram calculados as médias e desvios padrões dos resultados de cada método.

4.4 Análise dos resultados

Após o treinamento das RNAs e execução dos algoritmos de seleção de características, foi feita uma análise comparativa dos resultados confrontando a importância das variáveis de entrada estipuladas por cada método com os obtidos utilizando Correlação Linear.

A Correlação Linear mede a "força" ou "grau" de relacionamento entre duas variáveis. A medida da Correlação Linear (Equação 13), denominada coeficiente de correlação, varia no intervalo de -1 a 1 sendo que: -1 indica correlação negativa perfeita; 0 ausência de correlação; e 1 correlação positiva perfeita.

$$r = \frac{n(\sum xy) - (\sum x) \times (\sum y)}{\sqrt{n(\sum x^2) - (\sum x)^2} \times \sqrt{n(\sum y^2) - (\sum y)^2}} \quad (13)$$

em que r é o coeficiente de correlação, x e y são as variáveis que se deseja medir a correlação e n é o total de amostras.

5. RESULTADOS

Serão apresentados nesta seção os resultados obtidos para cada problema selecionado.

5.1 Íris

Esta seção demonstra os resultados obtidos pelos métodos de seleção de características para o problema Íris. Após o treinamento das RNAs os métodos de seleção de características foram executados apresentando os resultados exibidos na Tabela 1.

Para o problema da Íris, todos os métodos apontaram as variáveis **3** e **4** (Comprimento das pétalas e Largura das pétalas) como as mais importantes, concordando com a Correlação Linear. As variáveis menos importantes foram: a variável **2** (Largura das sépalas) de acordo com a Correlação Linear; e **1** (Comprimento das sépalas) para os demais métodos. Vale lembrar que o sinal negativo da Correlação Linear apenas indica uma inversão da variação da saída em relação à entrada testada.

Tabela 1. Importância dos atributos de acordo com cada método de seleção de características para o problema Íris.

Atributo	Correlação	Garson	Perturb	Sensibilidade	PaD
1	0,7283	11,3%	0,00952	$3{,}597 \times 10^{-5}$	0,06468
2	- 0,6840	23,7%	0,01086	$17{,}971 \times 10^{-5}$	0,23410
3	**0,9700**	**31,6%**	**0,01370**	**$33{,}394 \times 10^{-5}$**	**0,41927**
4	**0,9602**	**29,2%**	**0,01702**	**$30{,}75 \times 10^{-5}$**	**0,40772**

5.2 Desempenho da CPU

Esta seção demonstra os resultados obtidos pelos métodos de seleção de características para o problema Desempenho da CPU. Após o treinamento das RNAs, os métodos de seleção de características foram executados apresentando os resultados exibidos na Tabela 2.

Para o problema Desempenho da CPU, o atributo mais importante de acordo com os métodos Garson, Sensibilidade e PaD foi o **3** (maximum main memory in kilobytes), concordando com a Correlação Linear. O método Perturb identificou a variável **1** como sendo a mais importante. A variável considerada menos importante pelos métodos de seleção foi a **5** (minimum channels in units), enquanto a variável **1** (machine cycle time in nanoseconds) foi a identificada menos importante pela Correlação Linear.

Tabela 2. Importância dos atributos de acordo com os métodos apresentados para o problema Desempenho da CPU.

Atributo	Correlação	Garson	Perturb	Sensibilidade	PaD
1	-0,3071	14,9%	**0,01855**	0,005037	1,2501
2	0,7949	20,5%	0,01502	0,013737	2,7984
3	**0,8630**	**28,9%**	0,01592	**0,023963**	**5,9233**
4	0,6626	13,4%	0,01761	0,004184	0,8869
5	0,6089	9,3%	0,01402	0,001002	0,2805
6	0,6052	13,0%	0,01689	0,004199	1,2549

5.3 Resistência do Concreto

Esta seção demonstra os resultados obtidos pelos métodos de seleção de características para o problema da Resistência do Concreto. Após o treinamento das RNAs, os métodos foram executados apresentando os resultados exibidos na Tabela 3.

Para o problema da Resistência do Concreto, a variável mais importante de acordo com o algoritmo de Garson e a Correlação Linear foi a variável **1** (Cimento). Para o método Perturb a variável mais importante foi a **3** (resíduo da combustão). Para os métodos de Sensibilidade e PaD, a variável mais importante é a **8** (Idade). A variável considerada menos importante pelo algoritmo de Garson e o método Perturb foi a **4** (Água). A variável **6** (Agregado graúdo) foi identificada como menos importante pela Análise de Sensibilidade e PaD, contrariando a Correlação Linear que identificou a variável **3** (resíduo da combustão) como sendo a menos importante (apenas o módulo do valor da correlação interessa aqui).

Tabela 3. Importância dos atributos de acordo com cada método de seleção de características para o problema da Resistência do Concreto.

Atributo	Correlação	Garson	Perturb	Sensibilidade	PaD
1	**0,4978**	**15,7 %**	0,7097	10,3901	2705,83
2	0,1348	12,0 %	1,0642	2,46488	778,526
3	-0,1058	15,2 %	**1,4395**	5,25799	1711,86
4	-0,2896	9,0 %	0,6436	2,97607	1299,15
5	0,3661	13,2%	1,0109	5,11057	1913,59
6	-0,1649	9,8 %	0,8862	0,95426	316,85
7	-0,1672	15,4%	0,9954	6,20653	2462,41
8	0,3289	13,0%	0,8114	**12,468**	**3104,60**

6. CONCLUSÃO

O objetivo principal deste trabalho foi implementar e testar os principais métodos de seleção de características de dados em Redes Neurais Artificiais encontrados na literatura, visando identificar a contribuição de cada variável de entrada em relação a saída. O uso de algoritmos baseados nos pesos e sensibilidades (derivadas parciais) obtidos a partir do treinamento de uma RNA são uma boa alternativa aos métodos estatísticos tradicionais como se pode observar pelos resultados encontrados para diferentes problemas.

Para problemas de natureza mais simples, como o da Íris por exemplo, todos os métodos apresentaram resultados satisfatórios, identificando claramente as variáveis mais importantes. O algoritmo de Garson apresentou um desempenho satisfatório na determinação da importância das variáveis em todos os problemas quando comparado com os resultados obtidos por Correlação Linear. Os métodos de Análise de Sensibilidade e PaD apresentaram resultados praticamente iguais em todos os problemas devido ao fato de utilizarem o mesmo conceito (derivadas parciais) em sua implementação. Ambos se destacaram em relação aos demais justamente por identificar com maior precisão tanto as variáveis mais importantes quanto as menos significativas. Entretanto, apresentaram discrepâncias em relação à Correlação Linear.

Os métodos de seleção de características utilizados neste trabalho, ao identificar a importância das variáveis, possibilitam a eliminação das variáveis menos significativas reduzindo-se o número de entradas da rede neural. Isto permite que o treinamento da RNA seja mais rápido e ameniza o problema da maldição da dimensionalidade sem que a rede neural perca seu desempenho.

7. AGRADECIMENTOS

Os autores agradecem o apoio da FAPEMIG - Fundação de Amparo a Pesquisa do Estado de Minas Gerais - pelo apoio financeiro para publicação deste artigo.

8. REFERÊNCIAS

[1] Barreto, J. M. 1997. Introdução as Redes Neurais Artificiais. V Escola Regional de Informática da SBC Regional Sul (Santa Maria, Florianópolis, Maringá, 5-10 de maio de 1997), 41-71.

[2] Braga, A. P.; Carvalho, A.; Ludermir, T. 2000. Redes Neurais Artificiais: Teoria e Aplicações. Editora LTC, Rio de Janeiro, 262p.

[3] Caloba, G. M; Caloba, L. P; Saliby, E. 2002. Cooperação entre redes neurais artificiais e técnicas "clássicas" para previsão de demanda de uma série de vendas de cerveja na Austrália. Pesquisa Operacional, Rio de Janeiro, v. 22, n. 3 (julho 2002), 345-358. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0101-74382002000300004&lng=en&nrm=iso>. Acesso em: 15 Set. 2016.

[4] Cybenko, G. 1989. Approximations by superpositions of sigmoidal functions. Mathematics of Control Signals and Systems, v. 2 (1989), 303-314.

[5] Dimopoulos, Y.; Bourret, P.; Lek, S. 1995. Use of some sensitivity criteria for choosing networks with good generalization ability. Neural Processing Letters, v. 2, i. 6, (December 1995), 1-4. DOI: 10.1007/BF02309007

[6] Dimopoulos, I.; Chronopoulos, J.; Chronopoulou-Sereli, A.; Lek, S. 1999. Neural network models to study relationships between lead concentration in grasses and permanent urban descriptors in Athens city (Greece). Ecological Modelling, v. 120 (August 1999), 157-165.

[7] Frank, A; Assuncion, A. 2011. UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA. Disponível em: <http://archive.ics.uci.edu/ml>. Acessado em: 10/10/2016.

[8] Garson, G. D. 1991. Interpreting neural network connection weights. Artificial Intelligence Expert, v. 6, i. 4 (April 1991), 46-51, Miller Freeman, Inc. San Francisco, CA, USA.

[9] Grevey, M; Dimopoulos, I; Lek, S. 2003. Review and comparison of methods to study the contribution of variables in artificial neural network models. Ecological Modelling, v. 160, i. 3 (15 fevereiro 2003), 249-264.

[10] Haykin, S. 1998. Neural Networks: A Comprehensive Foundation. 2nd ed. Prentice Hall, NJ, USA, 842p. ISBN: 0132733501

[11] Hassoun, M. H. 1995. Fundamentals of Artificial Neural Networks. 1st ed. MIT Press, 511p.

[12] Hristev, R. 2011. ANN Toolbox for Scilab. Disponível em: http://www.scilab.org/contrib/index_contrib.php?page=displayContribution&fileID=166. Acessado em: 20/10/2016.

[13] Kovacs, K. L. 2002. Redes Neurais Artificiais - Fundamentos e Aplicações. 4ª ed. Livraria da Física, São Paulo, 174p.

[14] McCulloch, W. S.; Pitts, W. 1943. A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics, v. 5, i. 4 (December 1943), 115-133. DOI: 10.1007/BF02478259

[15] Minsk, M.; Papert, S. 1969. Perceptrons: an introduction to computational geometry. MIT Press, Massachusetts, 308p. ISBN: 9780262631112

[16] Rosenblatt, F. 1958. The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, v. 65, i. 6 (November 1958), 386-408.

[17] Valença, M. J. S.; Ludermir, T. B. 2007. Explicando a relação entre as variáveis de uma rede neural: Iluminando a

"Caixa Preta". XVII Simpósio Brasileiro de Recursos Hídricos, São Paulo (25-29 novembro, 2007).

[18] Valiant, L. G. 1984. A theory of the learnable. Magazine Communications of the ACM, v. 27, n. 11 (November 1984), 1134-1142.

[19] Veelenturf, L. P. J. 1995. Analysis and Applications of Artificial Neural Networks. NJ, USA: Prentience-Hall, 1995. 259p.

[20] Widrow, B.; Hoff, M.E. 1960. Adaptative switching circuits. Institute of Radio Engineers, Western Electronic Show and Convention Record, v. 4 (1960), 96-104.

[21] Yao, J.; Teng, N.; Poh, H.;Tan, C. 1998. Forecasting and analysis of marketing data using neural networks. Journal of Information Science and Engineering, v. 14, n. 4, 843-862.