XXXII CILAMCE: Congresso Ibero Latino Americano de Métodos Computacionais em Engenharia
13 a 16 de novembro de 2011 - Ouro Preto - MG

UMA REDE NEURAL DE BASE RADIAL BASEADA EM COMPUTAÇÃO EVOLUCIONÁRIA

Juliano F. da Mota[a,c], **Paulo H. Siqueira**[b,c], **Luzia V. de Souza**[b,c] **e Adriano Vitor**[a,c]

[a]*Departamento de Matemática, Universidade Estadual do Paraná - Campus de Campo Mourão,87303-100, Campo Mourão, PR, Brasil, http://www.fecilcam.br*

[b]*Departamento de Expressão Gráfica, Universidade Federal do Paraná, Centro Politécnico, Jardim das Américas, 81531-970, Curitiba, PR, Brasil, http://www.degraf.ufpr.br*

[c]*Programa de Pós-Graduação em Métodos Numéricos em Engenharia, Universidade Federal do Paraná, Centro Politécnico, 81531-970, Curitiba, PR, Brasil, http://www.ppgmne.ufpr.br*

Palavras Chave: Redes Neurais de Base Radial, Computação Evolucionária, Classificação de Padrões.

Resumo. Um dos problemas da modelagem de uma RBFNN - Radial Basis Neural Network, Rede Neural de Base Radial, consiste em determinar os pesos da camada de saída, geralmente representados por uma matriz retangular. Um inconveniente nesta fase é a necessidade da pseudo-inversão da matriz com os valores de ativação da camada intermediária. Esta operação, nos casos em que a quantidade de variáveis é grande ou os valores de ativação formam uma matriz mal condicionada, pode se tornar computacionalmente custosa ou pode ocasionar erros de arredondamento, fazendo com que o modelo não consiga classificar corretamente os padrões. Nesta pesquisa, são utilizados Algoritmos Genéticos de variáveis contínuas para determinar os pesos da camada de saída de uma RBFNN e também é feita uma comparação com o desempenho do método tradicionalmente utilizado, que é o da pseudo-inversão. A abordagem proposta consiste em gerar matrizes de pesos aleatórios normalmente distribuídos, que são os indivíduos da população e aplicar os operadores genéticos de Michalewicz até que algum dos critérios de parada seja atingido. Foram testados quatro bancos de dados de classificação de padrões e os resultados apontam um acerto contido no intervalo 91–98%, no melhor caso e 58–63%, no pior caso.

1 INTRODUÇÃO

Há tempos os cientistas tentam desenvolver métodos para prever valores em problemas de classificação para uma melhor tomada de decisão em cenários de incerteza. Uma gama de métodos matemáticos e estatísticos já foi desenvolvida e, por este motivo, busca-se melhorar cada vez mais os métodos já existentes e, também, o desenvolvimento de novos métodos que reduzam o custo computacional, ofereçam melhores resultados no que diz respeito ao desempenho do modelo ou ambos.

Todos os métodos existentes já oferecem muitas opções quanto à velocidade e qualidade da previsão ou classificação. A tarefa mais difícil, que todo pesquisador anseia, é desenvolver um método que consiga elevada acurácia o mais rápido possível, dada a necessidade de rapidez na era *on-line* em que a sociedade se encontra. Por "elevada acurácia" entendemos o menor erro possível, num problema de previsão, e o maior percentual possível de classificações corretas, para o caso de um problema de classificação.

A idéia de criar um grupo de modelos matemáticos que simule os neurônios humanos é considerada, por muitos cientistas, brilhante e extremamente útil para os responsáveis pela tomada de decisão nas mais diversas áreas do conhecimento. Tais modelos são as Redes Neurais Artificiais. Segundo (Haykin, 2001), uma rede neural é uma máquina de processamento paralelo que pode transformar conhecimento experimental em informações úteis para utilização, por exemplo, no auxílio à tomada de decisão.

Utilizar Redes Neurais na classificação de padrões, pode constituir uma área de estudo interessante e possivelmente a combinação de tal técnica com algoritmos que fazem a combinação de preditores (*Boosting*) como podemos ver em (Souza et al., 2009), que tem mostrado bons resultados, possa ser melhorada alterando os modelos já existentes bem como criando novos modelos.

Esta pesquisa compara dois métodos de treinamento de uma Rede Neural de Base Radial, um é considerado tradicional, que se baseia na pseudo-inversão de uma matriz retangular e o outro utiliza Algoritmos Genéticos, convertendo esta matriz em indivíduos de uma população genética, buscando encontrar a matriz ótima (ou quase-ótima) através de seleção natural e operadores genéticos, mais especificamente os propostos por (Michalewicz et al., 1994).

Para tanto, preceitos básicos sobre Redes Neurais de Base Radial e Algoritmos Genéticos são apresentados e suas principais características são explicitadas, afim de formar uma base para os experimentos realizados.

2 REVISÃO DE LITERATURA

2.1 Redes Neurais de Base Radial

Uma Rede Neural de Base Radial (RBFNN) é alimentada adiante e possui apenas duas camadas, sendo uma a intermediária e a última a camada de saída. Na camada intermediária as funções de ativação dos neurônios são ditas *funções de base radial* e, por isso, antes de explicitarmos as demais características principais de uma RBFNN, a definição de função de base radial está apresentada a seguir.

2.1.1 Funções de Base Radial

Uma RBF - Radial Base Function (Função de Base Radial) é definida por (Haykin, 2001) como qualquer função que satisfaz a Equação 1. Dizemos que uma função é de base radial

quando seus valores funcionais são iguais às normas de seus argumentos.

$$f(x) = f(\|x\|) \tag{1}$$

Em outras palavras, uma função é de base radial quando seu valor funcional depende apenas da distância de seu argumento à origem. Há várias RBFs, alguns exemplos são apresentados a seguir.

$$f(x) = e^{-\beta x^2}, \text{para } \beta > 0 \tag{2}$$

$$f(x) = \sqrt{x^2 + \beta^2}, \text{para } \beta > 0 \tag{3}$$

$$f(x) = x^{2k+1}, \text{para } k = 0, 1, \ldots \tag{4}$$

$$f(x) = x^{2k}, \text{para } k = 0, 1, \ldots \tag{5}$$

De acordo com (Powell, 1988), a abordagem que busca aproximar funções por meio de combinações lineares de funções de base radial consiste em introduzir um conjunto de N funções base, uma para cada observação amostral, tomando a forma $\phi(\|x - x^n\|)$ em que $\phi(\ .\ )$ é uma RBF. Então, cada função base depende apenas da distância $(\|x - x^n\|)$ e a aproximação toma a forma definida pela Equação 6.

$$h(x) = \sum_{n=1}^{N} w_n \cdot \phi(\|x - x^n\|) \tag{6}$$

Considerando esta abordagem para resolver o problema de interpolação exata na Equação 7, em que $t(x^n)$ são os valores alvo e $h(x^n)$ são os resultados do modelo de interpolação temos,

$$h(x^n) = t(x^n). \tag{7}$$

Combinando as Equações 6 e 7, temos

$$\sum_{n=1}^{N} w_n \cdot \phi(\|x - x^n\|) = t, \tag{8}$$

que podemos representar matricialmente por,

$$\phi \cdot w = t, \tag{9}$$

em que $t = (t^1, t^2, \ldots, t^n)$, $w = (w_1, w_2, \ldots, w_n)$ e ϕ é uma matriz quadrada. A solução formal para 9 é dada na Equação 10 e o Teorema de Michelli (Michelli, 1986) garante que, para várias funções, a matriz ϕ é invertível.

$$w = \phi^{-1} \cdot t \tag{10}$$

2.1.2 Redes Neurais que utilizam Funções de Base Radial

Uma RBFNN é alimentada adiante e utiliza funções de base radial como função de ativação dos neurônios da camada escondida. O processo de aprendizagem desta rede tem suas bases na teoria da Programação Não Linear. A arquitetura de uma RBFNN é bastante simples, há apenas duas camadas além dos nós de entrada: uma camada escondida, que possui funções de base radial como funções de ativação e uma camada de saída que possui funções lineares como ativação, como é possível ver na Figura 1.

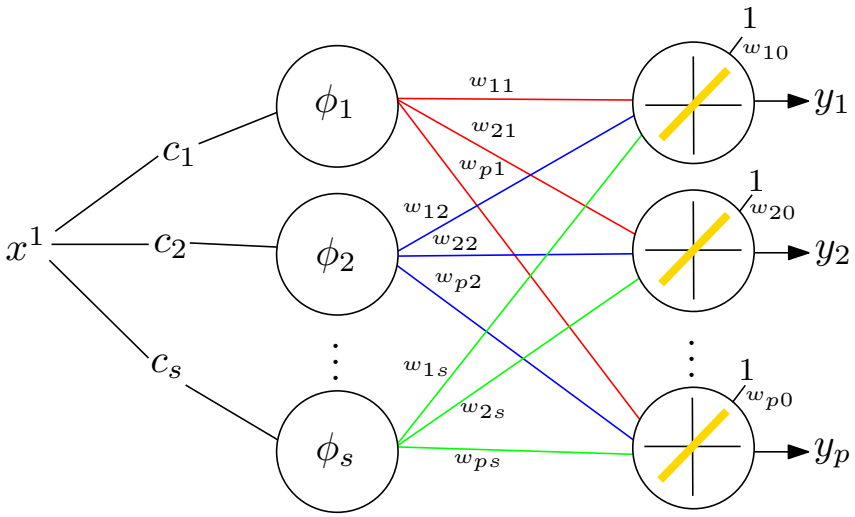

Figura 1: Representação da arquitetura de uma RBFNN

Cada neurônio da camada escondida possui um vetor associado, chamado de centro do neurônio, o qual define o centro do campo receptivo daquele neurônio. Geralmente tais vetores são armazenados numa matriz C, chamada de matriz de centros dos neurônios. Estes vetores exercem forte influência sobre o desempenho da rede. Posteriormente, na seção 2.1.3 alguns métodos serão brevemente apresentados.

Seja $X \subset \mathbb{R}^{m \times n}$ a matriz de observações contendo n observações e m variáveis, temos,

$$X = \begin{pmatrix} x_{11} & \dots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{m1} & \dots & x_{mn} \end{pmatrix}$$

O valor de ativação do m-ésimo neurônio da camada escondida depende da distância euclidiana quadrática entre a entrada x^i e o centro c_j, onde $c_j \in C$ é o centro do m-ésimo neurônio. Então, de acordo com a abordagem clássica, treinar uma RBFNN é, de fato, calcular a matriz de pesos,

$$w = \begin{pmatrix} w_{10} & w_{11} & \dots & w_{1s} \\ w_{20} & w_{21} & \dots & w_{2s} \\ \vdots & \ddots & \vdots & \vdots \\ w_{p0} & w_{p1} & \dots & w_{ps} \end{pmatrix}$$

de maneira a ajustar aos alvos t cada y, como podemos ver na Equação 11,

$$y_r = w_{0r}\phi_{0r} + \sum_{k=1}^{s} w_{kr}\phi\left(||x^i - C||\right). \tag{11}$$

Resolver o problema da Equação 11 é equivalente a resolver apresentado na Equação 9, a única diferença é a forma de ϕ, que no presente caso é retangular. Como é possível observar, treinar uma RBFNN é equivalente a resolver um sistema não linear retangular contendo funções de base radial.

2.1.3 Selecionando os Centros dos Neurônios

A tarefa de selecionar os centros dos neurônios é, essencialmente, um problema de agrupamento de padrões. Há algumas estratégias clássicas descritas em (Haykin, 2001) e uma breve discussão sobre estas estratégias é apresentada a seguir.

2.1.4 Centros Fixados Aleatoriamente

Esta é a maneira mais simples e menos custosa de selecionar os centros. Embora seja simples, é considerada por (Haykin, 2001) como a abordagem "mais sensata", pois a cada experimento uma parte diferente da matriz de observações é utilizada e, por este motivo, a matriz de centros passa a conter uma boa representação do espaço de entrada.

A limitação deste método reside na necessidade de um conjunto de treinamento grande para que um desempenho satisfatório seja alcançado. O padrão mais usado 60-20-20 de treinamento, ou seja, 60% para treinamento e 20% para validação e teste provavelmente teria de ser modificado e, consequentemente, o modelo poderia perder capacidade de generalização.

2.1.5 Seleção Auto-organizada de Centros

Também descrito em (Haykin, 2001) e (Silva et al., 2010), este método de seleção de centros consiste em aplicar a ideia básica do algoritmo SOM - *Self Organizing Map* (Mapa Auto-organizável), proposto por Kohonen *apud* (Haykin, 2001), o qual descrevemos a seguir:

1. Escolher valores randômicos (distintos) para os centros, geralmente tomando elementos do conjunto de observações como centros iniciais;
2. Durante a n-ésima iteração, tomar uma amostra do conjunto de treinamento;
3. Para cada vetor de entrada, encontrar o centro que possui menor distância euclidiana ou, equivalentemente, o centro com o qual o vetor de entrada produz o maior produto interno;
4. Para os centros vencedores, atualizar sua localização segundo a regra:

$$c(\text{novo}) = c(\text{velho}) + \eta(x - c) \tag{12}$$

   em que $\eta \in (0, 1)$ é uma taxa de aprendizagem, x é o vetor de entrada e c é o centro vencedor.
5. Voltar ao Passo 2 até que não haja modificações significativas na matriz de centros.

A diferença deste algoritmo para o SOM clássico é justamente a ausência de um "mapa" de neurônios, ou seja, nenhuma vizinhança é sequer considerada na atualização dos neurônios. Isso equivale a dizer que é um algoritmo SOM com o princípio *Winner Takes All*, O Vencedor Leva Tudo, somente o neurônio vencedor é atualizado. Uma descrição do princípio *Winner Takes All*, aplicado ao problema de designação usando uma rede recorrente, pode ser encontrado em (Siqueira et al., 2005).

2.1.6 Seleção Supervisionada de Centros

Esta é a forma mais genérica de se realizar a seleção de centros. A ideia é ajustar os neurônios na matriz de centros considerando aprendizagem por correção de erro e, desse modo, a RBFNN se assemelha ao *Perceptron* clássico. Um candidato natural para resolver este problema é o algoritmo conhecido como descida do gradiente, que é utilizado para minimizar uma função de custo/erro.

2.2 Algoritmos Genéticos para dados Contínuos

2.2.1 Princípios Fundamentais

Os AG - Algoritmos Genéticos conforme descrito em (Holland, 1975) *apud* (Man et al., 1996) são, basicamente, um conjunto de algoritmos baseados nos princípios de biologia evolutiva enunciados por Charles Darwin em seu livro mundialmente conhecido intituluado *A Evolução das Espécies* que buscam soluções aproximadas em problemas de otimização.

Os princípios considerados na elaboração de um AG são a seleção natural, hereditariedade, mutação e o recombinamento (*crossing-over*). Quando são traduzidos para a linguagem matemática, tais princípios se transformam em operadores genéticos de cruzamento e mutação, estes já levando em conta a hereditariedade e a seleção natural se transforma numa regra de decisão.

Figura 2: Algoritmo Genético Original

Seja $g = 0$ o contador de gerações;
Criar e inicializar uma população de tamanho pré-definido;
    Enquanto *nenhum critério de parada for satisfeito* faça;
        Avalie o *fitness* (aptidão) de cada indivíduo da população;
        Realize a reprodução para gerar descendência;
        Selecione a nova população;
        Avance para a nova geração, ou seja, $g = g + 1$;
    Fim Enquanto

O algoritmo básico, sistematizado por (Holland, 1975) *apud* (Man et al., 1996), possui baixa complexidade de implementação e essa é uma das grandes vantagens dessa técnica. As etapas do algoritmo estão explicidadas na Figura 2.

Vale ressaltar que, inicialmente, a base numérica binária (representação binária) foi considerada para implementação e aplicação dos Algoritmos Genéticos. Contudo, hoje é bastante comum encontrarmos aplicações em que a base numérica decimal (representação real) é utilizada. Possivelmente, essa mudança tenha ocorrido em virtude da representação real permitir maior variedade de operadores, como será apresentado na seção 2.2.2.

2.2.2 Representação Real e Operadores de Michalewicz *et al*

Uma das pesquisas que utilizam a representação real foi realizada por (Michalewicz et al., 1994) que em seu trabalho sistematizou três operadores de cruzamento (*Crossover*) e três operadores de mutação, os quais passamos a descrever. Adicionalmente, um operador de mutação a mais está descrito.

Crossover Simples

Este operador é uma variação do *crossover* convencional de um ponto, que é utilizado na representação binária, adaptado para a representação real, como é possível ver na Figura 3.

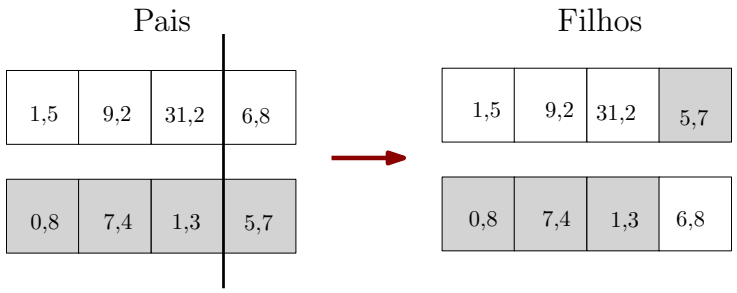

Figura 3: Representação do cruzamento genético de um ponto

Crossover Aritmético

Dados dois indivíduos p_1 e p_2 para o cruzamento, o *crossover* aritmético consiste em gerar os elementos,

$$c_1 = \beta p_1 + (1-\beta)p_2 \text{ e } c_2 = \beta p_2 + (1-\beta)p_1 \text{ com } \beta \sim U(0,1).$$

Vale ressaltar que este operador não ultrapassa os intervalos (p_1, p_2).

Crossover Heurístico

Dados dois indivíduos p_1 e p_2 para o cruzamento, tal que a aptidão de p_1 é maior do que a de p_2, este operador é a extrapolação linear entre os geradores,

$$c = p_1 + \beta(p_2 - p_1) \text{ onde } \beta \sim U(0,1),$$

garantindo que o *crossover* aritmético não leve os genes para o centro do intervalo.

Mutação Uniforme

Dado um indivíduo p, este operador substitui um gene por um número aleatório oriundo de uma distribuição uniforme, ou seja,

$$c_i = \begin{cases} U(a_i, b_i), & \text{se } i = j \\ p_i, & \text{caso contrário.} \end{cases}$$

Os valores a_i e b_i representam os limites do intervalo permitido para o indivíduo c_i em sua j-ésima componente, caso haja alguma restrição de factibilidade.

Mutação de Limite

Dado um indivíduo p, este operador substitui um gene por um dos limites do intervalo factível $[a_i, b_i]$, caso haja alguma restrição de factibilidade, evitando que o *crossover* aritmético leve os genes para o centro do intervalo factível, onde $r \sim U(0,1)$,

$$c_i = \begin{cases} a_i, & \text{se } r < 0,5 \text{ e } i = j \\ b_i, & \text{se } r \geqslant 0,5 \text{ e } i = j \\ c_i, & \text{caso contrário .} \end{cases}$$

Mutação Não-uniforme

Dado um indivíduo p, este operador substitui um gene por um número extraído de uma distribuição não-uniforme,

$$c_i = \begin{cases} p_i + (b_i - p_i)f(G), & \text{se } r_1 < 0,5 \text{ e } i = j \\ p_i - (p_i - a_i)f(G), & \text{se } r_1 \geq 0,5 \text{ e } i = j \\ p_i, & \text{caso contrário .} \end{cases}$$

com,

$$f(G) = \left[r_2 \left(1 - \frac{G}{G_{\max}} \right) \right]^b,$$

sendo G a geração atual, $G_{\max}$ o número máximo de gerações e $r_1, r_2 \sim U(0,1)$. Além disso, os valores a_i e b_i são os limites do intervalo factível.

Mutação Não-uniforme Múltipla

Dado um indivíduo p, a aplicação da mutação não-uniforme em todos os genes deste indivíduo é chamada de mutação não-uniforme múltipla.

2.3 Trabalhos Relacionados

Buscando solucionar o principal problema da abordagem clássica para o treinamento de uma RBFNN, que é a necessidade de calcular a pseudo-inversa de uma matriz retangular, alguns autores propuseram metodologias alternativas para trocar o referido problema por outro de menor complexidade computacional.

Os pesquisadores (Li e Ling, 2011) aplicaram a ideia de obter os pesos da camada intermediária numa RBFNN para desenvolver um modelo de controle preditivo generalizado numa unidade geradora de energia que possuía duas variáveis de entrada e duas variáveis de saída. O experimento realizado consistiu em comparar o desempenho do algoritmo proposto (chamado de GA-RBF em alusão aos Algoritmos Genéticos e à RBFNN) com o desempenho de um *Perceptron* de Múltiplas Camadas com a utilização do algoritmo de treinamento de retro-propagação do erro. A técnica proposta teve um erro quadrático médio da ordem de 10^{-5}, enquanto o Perceptron testado errou aproximadamente 10^{-1}.

O trabalho de (Changbing e Wei, 2010) aplicou uma RBFNN com treinamento via AG na previsão de risco num ambiente aquático. Os testes mostraram que o modelo proposto apresentou um desempenho superior ao "modelo cinza", que utiliza uma equação diferencial de primeira ordem para estimar valores com base me dados históricos. Há gráficos nos quais é possível vizualizar a superioridade do modelo proposto para a predição do risco.

Já (Ming et al., 2010) aplicaram basicamente a mesma ideia na predição do fluxo numa rede. Os resultados do experimento realizado, no qual o fluxo de uma rede com 40 pontos foi predito, mostrou que o erro relativo na predição de uma RBFNN com o treinamento via pseudo-inversão (tradicional) foi de $0,0247$ enquanto que a RBFNN com treinamento via AG conseguiu errar apenas $0,0177$, uma redução de quase 30% no erro.

A pesquisa de (Kurban e Beşdok, 2009) comparou quatro algoritmos para o treinamento de uma RBFNN: Colônia de Abelhas, Algoritmos Genéticos, Filtro de Kalman e Descida do Gradiente. Nos três bancos de dados testados disponíveis em (Frank e Asuncion, 2010) e no banco de dados da aplicação proposta pelos autores o algoritmo "Colônia de Abelhas" apresentou um desempenho levemente superior ao AG, ambos apresentando uma acurácia acima de 90% de acerto no conjunto de testes nos bancos de dados em (Frank e Asuncion, 2010) e acima de 70% no banco de dados da aplicação em sensores.

Uma abordagem relativamente interessante pode ser encontrada em (Zhangang et al., 2007), pois nesta pesquisa todos os parâmetros da RBFNN são convertidos em indivíduos (vetores) para que se aplique a sequencia de AG e, posteriormente, ao fim de cada geração, ordenadamente, os componentes deste indivíduo são designados a cada parâmetro da RBFNN tomando elementos do vetor de acordo com a quantidade de informações necessárias a cada parâmetro. O trabalho é relacionado à previsão da série temporal de carga elétrica numa cidade e o maior erro absoluto no conjunto de testes foi de 3,05%.

Um pouco mais conservadora é a pesquisa de (Jareanpon et al., 2004) em que o único parâmetro da RBFNN calculado via AG foi o raio de abertura da Equação 2. A cada mudança no modelo uma população inicial era criada e, posteriormente, a sequencia de passos característicos de um AG foram aplicados em bancos de dados relacionados a níveis de precipitação pluviométrica do departamento de Meteorologia da Tailândia, mais especificamente em 13 cidades, e os valores da soma quadrática dos erros na predição, em todos os casos, ficaram abaixo de $0,1$, sendo o menor deles $1,02 \cdot 10^{-21}$ e o maior deles $7,67 \cdot 10^{-2}$.

Anteriormente a todos os trabalhos mencionados, (Kuncheva, 1997) utilizou AG para a inicialização dos parâmetros de uma RBFNN e também testou dois conjuntos de dados disponíveis em (Frank e Asuncion, 2010). Tanto no primeiro quanto no segundo conjunto houve uma acurácia entre 65–70% em todas as configurações utilizadas, havendo uma leve superioridade do AG em comparação com outras técnicas.

3 EXPERIMENTOS E ANÁLISE DOS RESULTADOS

Nos experimentos realizados, o desempenho de uma RBFNN usando o método tradicional de treinamento, a pseudo-inversão da matriz que contém os valores de ativação da camada intermediária, foi comparado com o treinamento alternativo via AG, utilizando os operadores de Michalewicz (Michalewicz et al., 1994). Foram utilizados os já bastante conhecidos problemas de classificação - Iris, Contraceptive Method Choice (CMC), Cancer e Blood Transfusion (BT) - que são conjuntos de dados encontrados em (Frank e Asuncion, 2010).

Tabela 1: Características dos conjuntos de dados

Conjunto	Entradas	Saídas	Treinamento	Validação	Teste
Iris	4	3	90	30	30
CMC	9	3	884	295	294
Cancer	9	2	420	140	139
BT	4	2	449	150	149

Para todos os conjuntos de dados, foi realizado um experimento que consistiu em cinco etapas:

1. separar o banco de dados em três conjuntos:

- conjunto de treinamento, com 60% dos exemplos;
- conjunto de validação, com 20% dos exemplos e;
- conjunto de testes, com 20% dos exemplos.

2. treinar uma RBFNN 100 vezes com cada abordagem (pseudo-inversão e AG) e obter o PCC - Percentual de Classificação Correta em cada conjunto do item 1, especialmente o conjunto de testes;

3. registrar os resultados estatísticos das porcentagens de classificações corretas dos dois métodos de treinamento;

4. considerando um nível de significância de 5%, testar a hipótese de igualdade das variâncias afim de determinar qual teste de comparação de médias utilizar, fazendo $H = 0$ caso as variâncias sejam iguais e $H = 1$, caso contrário;

5. considerando um nível de significância de 5%, realizar dois testes de comparação de médias com as seguintes hipóteses:

Primeiro Teste	Segundo Teste
h_0 : Média PI = Média AG	h_0 : Média PI = Média AG
h_1 : Média PI > Média AG	h_2 : Média PI < Média AG

   fazendo $h_1 = 0$ se h_0 não for rejeitada no primeiro teste e $h_1 = 1$, caso contrário. E, ainda, fazendo $h_2 = 0$ se h_0 não for rejeitada no segundo teste e $h_2 = 1$, caso contrário.

As características dos conjuntos utilizados nos testes estão na Tabela 1, a quantidade de variáveis nos padrões de entrada, saída e a quantidade de padrões em cada conjunto.

Vale salientar que o conjunto de validação serve apenas para orientar o treinamento da rede neural no sentido de avaliar sua capacidade de generalização. A única informação do conjunto de validação utilizada no treinamento é o percentual de classificações corretas realizadas pela rede neste conjunto. A estratégia comumente adotada é a seguinte: enquanto o percentual de classificações corretas no conjunto de validação estiver aumentando a rede deve continuar seu treinamento, após haver quedas consecutivas é recomendado parar o treinamento e escolher a melhor configuração encontrada até então.

Já do conjunto de testes, que mede o desempenho efetivamente alcançado, nenhuma informação é utilizada durante o treinamento da rede em questão. Este conjunto serve somente para a avaliação final da técnica aplicada e informa uma probabilidade de acerto do modelo desenvolvido.

De acordo com a literatura, um dos parâmetros mais importantes para uma rede neural, inclusive as RBFNN é a quantidade de neurônios em suas camadas, no caso específico desta pesquisa, a quantidade de neurônios da camada intermediária. O limite aqui estabelecido será o de testar de dois a 10 neurônios, com incremento de dois a cada experimento.

Outras duas grandes questões, no que diz respeito aos parâmetros de uma RBFNN são: o método de seleção dos centros dos neurônios e a definição do raio de abertura das funções de ativação gaussianas na camada intermediária. O método de seleção de centros utilizado nesta pesquisa foi a seleção auto-organizada de centros, descrito na Seção 2.1.5.

Já o método para definição do raio de abertura de cada RBF - *Radial Basis Function*, Função de Base Radial, está descrito em (Silva et al., 2010) e consiste, basicamente, em calcular o

raio de cada função com base na distância quadrática média entre as entradas e os centros dos neurônios que receberam alguma atualização durante a seleção auto-organizada dos centros.

A parametrização de um AG também não possui um método consolidado, variando de acordo com cada aplicação. Por esse motivo, faz-se necessária a realização de alguns testes iniciais para que se possa observar a influência de cada parâmetro no desempenho do algoritmo. Nesta pesquisa, foram realizados mais de 50 testes preliminares e a melhor parametrização encontrada está na Tabela 2.

Com relação aos operadores de mutação que possuem restrição no intervalo dos números gerados, ou seja, a Mutação de Limite, Mutação Não-uniforme e Mutação Não-uniforme Múltipla, o intervalo adotado em todas as situações foi o de aceitar um aumento de até 100% do maior valor (em módulo) em qualquer gene de qualquer indivíduo.

Isso quer dizer que se o maior valor (em módulo) dentre os genes de todos os indivíduos fosse τ, então, o intervalo possível para geração de um gene para mutação seria $(-2 \cdot \tau, 2 \cdot \tau)$. Essa medida buscou um equilíbrio entre a extrapolação excessiva do espaço de busca e a restrição excessiva do espaço de busca.

Tabela 2: Tabela dos parâmetros utilizados para o AG

Parâmetro	Valor
População Inicial	100
Probabilidade Operadores Cruzamento	80%
Probabilidade Operadores Mutação	10%
Número Máximo de Gerações	100

Os resultados estatísticos das 100 rodadas para cada conjunto de dados estão nas Tabelas 3–6 para os conjuntos Iris, CMC, Cancer e BT, respectivamente. Nas tabelas é possível observar as informações sobre a média e desvio-padrão do desempenho de cada algoritmo para cada conjunto de dados. Nas tabelas, a variável "NNeuro" representa a quantidade de neurônios na camada escondida. Para cada quantidade de neurônios, na primeira linha estão representadas as estatísticas do método que utiliza a pseudo-inversão e na segunda a abordagem via AG.

Tabela 3: Resultados para o conjunto Iris

NNeuro	Média $\pm$ Desvio-padrão	Mínimo	Máximo	H	h_1	h_2
2	$90,5\% \pm 5,6\%$	73,3%	100%	1	0	0
	$88,4\% \pm 7,8\%$	63,3%	100%			
4	$92,3\% \pm 5\%$	73,3%	100%	1	1	0
	$87,3\% \pm 7\%$	66,7%	96,7%			
6	$93,9\% \pm 3,9\%$	86,7%	100%	1	1	0
	$88,7\% \pm 5,6\%$	73,3%	100%			
8	$94,6\% \pm 3,8\%$	83,3%	100%	1	1	0
	$87,1\% \pm 7,3\%$	70%	100%			
10	$94,6\% \pm 3,8\%$	86,7%	100%	1	1	0
	$85,2\% \pm 8,1\%$	63,3%	100%			

Como é possível observar na Tabela 3, o modelo proposto que utiliza AG para obtenção da matriz de pesos apresentou uma variabilidade ligeiramente maior do que a abordagem via pseudo-inversão. O PCC médio da abordagem via AG não foi superior à pseudo-inversão em

nenhum caso e de acordo com o teste *t*, para comparação entre duas médias, houve empate com dois neurônios e superioridade da abordagem via pseudo-inversão com as outras quantidades de neurônios testadas, sendo que em todos os testes a significância foi de 5%.

Nos resultados da Tabela 4, é possível observar que, considerando-se uma significância de 5%, a variância dos desempenhos de ambas as técnicas não tem diferença estatisticamente significativa e há superioridade da abordagem via pseudo-inversão em dois casos: para seis e 10 neurônios na camada intermediária, tendo em vista os valores de h_1. Mesmo comparando o melhor desempenho nos dois casos, a abordagem via AG se mostrou inferior dois pontos percentuais, aproximadamente.

Tabela 4: Resultados para o conjunto Contraceptive Method Choice

NNeuro	Média ± Desvio-padrão	Mínimo	Máximo	H	h_1	h_2
2	$57\% \pm 2,7\%$ $56,7\% \pm 2,9\%$	49% 49%	63,3% 63,3%	0	0	0
4	$57,1\% \pm 2,2\%$ $56,7\% \pm 2,6\%$	52,7% 50,7%	61,2% 62,6%	0	0	0
6	$58,5\% \pm 3,3\%$ $56,9\% \pm 3,2\%$	51,7% 49,7%	67,7% 64,3%	0	1	0
8	$58,2\% \pm 3,2\%$ $57,1\% \pm 3,2\%$	50,7% 48,6%	65,6% 62,9%	0	0	0
10	$60,5\% \pm 2,8\%$ $58,4\% \pm 2,7\%$	52% 51,4%	65% 64%	0	1	0

No caso do conjunto Cancer, na Tabela 5, é possível notar que, considerando uma significância de 5%, não houve diferença entre as variâncias dos desempenhos de ambas as técnicas, exceto com 10 neurônios, e que no único caso em que houve diferença estatisticamente significativa, com seis neurônios na caomada escondida, esta foi favorável à abordagem via AG, tendo em vista o valor de h_2.

Tabela 5: Resultados para o conjunto Cancer

NNeuro	Média ± Desvio-padrão	Mínimo	Máximo	H	h_1	h_2
2	$95,6\% \pm 1,5\%$ $96\% \pm 1,5\%$	92,1% 92,9%	98,6% 98,6%	0	0	0
4	$95,3\% \pm 1,7\%$ $95,9\% \pm 1,4\%$	92,1% 92,9%	98,6% 98,6%	0	0	0
6	$95,1\% \pm 1,7\%$ $96\% \pm 1,6\%$	90,7% 92,1%	97,9% 98,6%	0	0	1
8	$96\% \pm 1,2\%$ $96,3\% \pm 1,3\%$	94,3% 93,6%	98,6% 98,6%	0	0	0
10	$95,8\% \pm 1,6\%$ $96,2\% \pm 1,4\%$	92,1% 93,6%	100% 99,3%	1	0	0

Finalmente, na Tabela 6, considerando 5% de significância, não houve diferença entre as variâncias dos desempenhos e a abordemboraagem via pseudo-inversão foi superior em três das cinco quantidades de neurônios testadas, sendo seis, oito e 10 neurônios. Para as quantidades dois e quatro neurônios houve empate.

Tabela 6: Resultados para o conjunto Blood Transfusion

NNeuro	Média ± Desvio-padrão	Mínimo	Máximo	H	h_1	h_2
2	$75,9\% \pm 3,2\%$	68,5%	83,9%	0	0	0
	$75,9\% \pm 3,2\%$	68,5%	83,9%			
4	$77,7\% \pm 3,3\%$	70,5%	84,6%	0	0	0
	$77,2\% \pm 3,2\%$	71,1%	85,2%			
6	$78,8\% \pm 3,9\%$	67,8%	89,3%	0	1	0
	$77,5\% \pm 3,4\%$	67,8%	85,2%			
8	$77,5\% \pm 2,4\%$	71,1%	81,9%	0	1	0
	$75,9\% \pm 3\%$	67,8%	81,9%			
10	$77,6\% \pm 3,5\%$	71,8%	85,2%	0	1	0
	$76\% \pm 3,5\%$	68,5%	84,6%			

4 CONSIDERAÇÕES FINAIS

4.1 Conclusões

Nesta pesquisa, os Algoritmos Genéticos foram utilizados para treinar uma Rede Neural de Base Radial e a abordagem tradicionalmente utilizada, que é a pseudo-inversão da matriz que contém os valores de ativação, foi utilizada como parâmetro para medir o desempenho do AG em problemas de classificação. Para tanto, foram utilizados conjuntos de dados do conhecido repositório de dados para aprendizado de máquina da UCI *Machine Learning Dataset* (Banco de dados para Aprendizado de Máquina da UCI) (Frank e Asuncion, 2010).

Treinar uma RBFNN consiste basicamente em definir: o posicionamento dos centros dos neurônios, o grau de abertura do raio das funções gaussianas de ativação e a matriz de pesos entre a camada escondida e a camada de saída. Ainda um parâmetro que é obtido experimentalmente, mas não é menos importante, é a quantidade de neurônios na camada escondida.

Considerando os resultados obtidos e analisados anteriormente, é possível afirmar que:

- os operadores genéticos apresentados não produziram boas populações genéticas para a aplicação no treinamento de RBFNNs, tendo em vista que não superaram a abordagem tradicional nos testes realizados, exceto num único caso que pode ser visto na Tabela 5;
- a *performance* do algoritmo AG em comparação com a abordagem tradicional deixou a desejar, pois mesmo na situação em que foi melhor, tal superioridade não se mostrou inquestionável.
- houve dois casos em que nenhuma das duas abordagens conseguiu um resultado expressivo: no conjunto de dados "*Contraceptive Method Choice*", os melhores desempenhos de ambos os algoritmos ficou abaixo dos 70% em todas as execuções e abaixo dos 60% na média. Já no caso do conjunto "*Blood Transfusion*", embora o PCC médio tenha sido maior que 70%, numa eventual aplicação na realidade não seria viável confiar no modelo;
- o treinamento via AG produziu resultados inferiores à pseudo-inversão em três dos quatro conjuntos testados, considerando a significância definida nos testes.

Tomando um apanhado geral dos resultados é possível afirmar que, considerando um nível de 5% de significância, a abordagem via AG, mesmo sendo capaz de gerar resultados tão bons quanto a abordagem supostamente exata, teve um desempenho estatisticamente inferior

nos bancos de dados testados. Entretanto é necessário ponderar que a abordagem via pseudo-inversão, com o crescimento da quantidade de neurônios na camada escondida, sofre com a complexidade numérica e os erros de arredondamento. Isso deixa espaço para que a característica adaptativa dos Algoritmos Genéticos, caso sejam desenvolvidos operadores genéticos mais robustos, venha a superar essa limitação.

4.2 Sugestões Para Trabalhos Futuros

Constatando a evidente insuficiência das considerações realizadas nesta pesquisa, abaixo estão listadas algumas possíveis sugestões para trabalhos que busquem continuar ou refutar esta linha de pensamento:

- testar outros operadores para AG contínuos;
- comparar outros algoritmos de busca (além de AG) com a abordagem via pseudo-inversão;
- comparar algoritmos de busca com a abordagem via aprendizagem supervisionada;
- utilizar técnicas estatísticas tais como a Análise de Componentes Principais e a Análise Fatorial no tratamento dos dados, afim de verificar se podem ajudar na melhoria do desempenho em conjuntos com características similares ao "*Contraceptive Method Choice*", na qual ambas as abordagens apresentaram desempenho inferior à 70%.

AGRADECIMENTOS

À Fundação Araucária - Fomento à Pesquisa Paranaense, pelo apoio financeiro concedido à pesquisa através do Programa de Capacitação Docente de universidades do Estado do Paraná, por meio da chamada 01/2009 e o convênio n^o 239/2009.

REFERÊNCIAS

Changbing L. e Wei H. Application of genetic algorithm-rbf neural network in water environment risk prediction. *2nd International Conference on Computer Engineering and Technology 2010*, p. 239–242, 2010.

Frank A. e Asuncion A. Uci machine learning repository. 2010.

Haykin S. *Redes neurais - princípios e práticas*. Bookman, 2001.

Holland J. *Adaption in Natural and Artificial Systems*. MIT Press, 1975.

Jareanpon C., Pensuwon W., Frank R.J., e Davey N. An adaptive rbf network optimised using a genetic algorithm applied to rainfall forecasting. *International Simposium on Communications and Information Technologies 2004*, p. 1005–1010, 2004.

Kuncheva L.I. Initializing of an rbf network by a genetic algorithm. *Neurocomputing*, 14:273–288, 1997.

Kurban T. e Beşdok E. A comparsion of rbf neural netowrk training algorithms for inertial sensor based terrian classification. *Sensors*, p. 6312–6329, 2009.

Li N. e Ling H. Study of an algorithm of ga-rbf neural network generalized predictive control for generating unit. *International Conference on Electric Information and Control Engineering 2011*, p. 1723–1726, 2011.

Man K., Tang K., e Kwong S. Genetic algorithms: concepts and applications. *IEEE Transactions on Industrial Electronics*, 43(5):519–534, 1996.

Michalewicz Z., Logan T.D., e Swaminathan S. Evolutionary operators for continuous convex parameter spaces. *Proceedings of the 3rd Annual Conference on Evolutionary Programming*, p. 84–97, 1994.

Michelli C.A. Interpolation of scattered data: distance matrices and conditionally positive definite functions. *Constructive Approximation*, 2:11–22, 1986.

Ming Z.Y., Bin Z.Y., e Zhong L.L. Application of genetic algorithm and rbf neural network in network flow prediction. *3rd IEEE International Conference on Computer Science and Information Technology 2010*, p. 298–301, 2010.

Powell M.J.D. Radial basis function approximation to polynomials. *Numerical Analysis 1987 Proceedings*, p. 223–241, 1988.

Silva I.N., Spatti D.H., e Flauzino R.A. *Redes neurais artificiais - para engenharia e ciências aplicadas*. Artliber, 2010.

Siqueira P., Scheer S., e Steiner M.T.A. Application of the "winner takes all"principle in wang's recurrent neural network for the assignment problem. *Lecture Notes in Computer Science*, 3496(1):731–738, 2005.

Souza L.V., Pozo A., Rosa J.M.C., e Neto A.C. Applying correlation to enhance boosting technique using genetic programming as base learner. *Applied Intelligence*, 33(3):1–11, 2009.

Zhangang Y., Yanbo C., e Cheng K.W.E. Genetic algorithm-based rbf neural network load forecasting model. *Power Engineering Society General Meeting*, p. 1–6, 2007.