DISSERTAÇÃO DE MESTRADO Nº 727

ABORDAGENS MULTI-OBJETIVO PARA O TREINAMENTO DE REDES NEURAIS E SELEÇÃO DE CARACTERÍSTICAS

Honovan Paz Rocha

DATA DA DEFESA: 02/03/2012

Universidade Federal de Minas Gerais

Escola de Engenharia

Programa de Pós-Graduação em Engenharia Elétrica

ABORDAGENS MULTI-OBJETIVO PARA O TREINAMENTO DE REDES NEURAIS E SELEÇÃO DE CARACTERÍSTICAS

Honovan Paz Rocha

Dissertação de Mestrado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do Título de Mestre em Engenharia Elétrica.

Orientador: Prof. Antônio de Pádua Braga

Belo Horizonte - MG

Março de 2012

"Abordagens Multi-objetivo para o Treinamento de Redes Neurais e Seleção de Características"

Honovan Paz Rocha

Dissertação de Mestrado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Mestre em Engenharia Elétrica.

Aprovada em 02 de março de 2012.

Por:

Prof. Dr. Antônio de Pádua Braga
DELT (UFMG) - Orientador

Prof. Dr. Marcelo Azevedo Costa
DEST (UFMG) – Co-Orientador

Prof. Dr. Thiago de Souza Rodrigues
DECOM (CEFET/MG)

Prof. Dr. Rodney Rezende Saldanha
DEE (UFMG)

Agradecimentos

- À Deus que sempre providenciou tudo em minha vida. Em quem creio e confio plenamente, que me conduziu até aqui e me levará muito mais além.
- Aos meus pais que sempre se preocuparam e se esforçaram para me dar uma educação de qualidade e à minha irmã por todo o carinho e compreensão. Amo vocês!
- Ao Professor Braga meu orientador, pela oportunidade, ajuda, paciência e entusiasmo no dia-a-dia, além de todo o conhecimento e experiências transmitidas que contribuiram fundamentalmente para o meu amadurecimento.
- Ao Professor Marcelo Azevedo Costa meu co-orientador, pela paciência, entusiasmo e grande auxílio na construção deste trabalho.
- Aos amigos do LITC e CPDEE, por toda ajuda que me deram.

Resumo

As redes neurais artificiais têm sido aplicadas com sucesso na resolução de problemas como aproximação de funções e classificação de padrões, onde a extração de um modelo pode ser de difícil visualização. A busca por um modelo que melhor represente o problema torna a habilidade de generalização a principal preocupação no treinamento de redes neurais artificiais, tarefa esta que se torna ainda mais difícil em ambientes com grande dimensionalidade. Neste contexto o presente trabalho propõe novas técnicas para o treinamento multi-objetivo de redes neurais, onde a minimização do risco e o controle de complexidade são os objetivos a serem atingidos com o treinamento de forma que se obtenha um modelo mais compatível ao problema. É proposta também uma abordagem à redução de dimensionalidade através da tarefa de seleção de características, em que os objetivos são diminuir a quantidade de atributos do problema e maximizar a taxa de classificação correta, tornando menos árdua a tarefa de classificadores em ambientes com grande número de dimensões.

Abstract

Artificial neural networks have been successfully applied in solving problems such as functions approximation and patterns classification, where the extraction of a model can be difficult to see. The search for a model that best represents the problem makes the generalization ability the main concern in the training of artificial neural networks, a task that becomes even more difficult in environments with large dimensionality. In this context, this paper proposes new techniques for training multi-objective neural network, where the minimization of the risk and the control of complexity are objectives to be achieved through of the training in order to obtain a model more compatible to the problem. It also proposed an approach to dimensionality reduction through the task of feature selection, in which the objectives are to reduce the number of attributes of the problem and maximize the correct classification rate, making it less arduous task of classifiers in environments with large numbers of dimensions.

Sumário

Lista de Abreviações

MLP	-	*Multi-Layer Perceptron*
MOBJ	-	Multi-objetivo
DEANN	-	Algoritmo para treinamento de rede utilizando evolução diferencial
NSS	-	*Norm Surface Search*
WD	-	Weight Decay
DE	-	Differential Evolution
CLONAL	-	Algoritmo de seleção clonal
KNN	-	*k-nearest-neighbor*
EQM	-	Erro médio quadrático
TME	-	Tempo médio de execução
ALL	-	*acute lymphoblastic leukemia*
AML	-	*acute myeloid leukemia*
RNA	-	Redes neurais artificiais

Lista de Símbolos

$\mathbf{w}$	-	Vetor de pesos da rede neural artificial
$\|w\|$	-	Norma do vetor de pesos da rede neural artificial
$\mathbf{e}$	-	Vetor de erros da rede neural artificial
$J(w)$	-	Erro médio quadrático obtido pela rede neural artificial
$\mathbf{X}$	-	Vetor dos padrões de entrada
d	-	Saída esperada da rede neural artificial
y	-	Saída real da RNA
w_0	-	*Bias* da RNA

Lista de Figuras

Lista de Tabelas

CAPÍTULO
1

Introdução

O treinamento de uma rede neural tem o objetivo de gerar um modelo que represente bem um determinado problema. Os fatores determinantes para que a rede aprenda uma determinada tarefa e obtenha um modelo compatível para representação desta requerem um grande esforço por parte do projetista na definição de parâmetros apropriados e na escolha de uma arquitetura adequada.

Uma rede neural bem treinada tem a capacidade de modelar o problema de forma que se reconheçam padrões desconhecidos ao processo de treinamento. Esta habilidade é definida como capacidade de generalização da rede.

Com base na teoria do aprendizado estatístico, onde se fundamenta o aprendizado de máquina e, consequentemente o treinamento de redes neurais, existe a necessidade de se aproximar à função geradora dos dados, desta maneira obtendo um modelo que represente bem a tarefa que se quer aprender. Este objetivo se torna uma tarefa difícil pelo fato de que se tem apenas uma amostragem do problema e não se conhece a função de densidade de probabilidade que gerou esta amostragem. Sendo assim o processo de aprendizagem tem como alvo a aproximação de um modelo de regressão [19] definido através de uma função determinística dos dados amostrados com o acréscimo de um erro, que representa o desconhecimento da função geradora da amostra.

Encontrar o modelo que minimize o desvio entre o modelo gerado pela rede e a função geradora dos dados requer a utilização de uma função de perda que deve ser minimizada no processo de aprendizagem. Esta função de perda pode ser utilizada para conduzir à minimização do risco funcional, que é dependente da função de densidade de probabilidade conjunta das entradas e

saída do problema, sendo geralmente desconhecida. Desta maneira faz-se necessário a minimização do risco empírico, que é definido em função dos dados amostrados para o treinamento da rede. Para que a utilização desta medida seja consistente no aprendizado, diversos fatores devem ser levados em consideração. O conjunto de dados de treinamento deve ser grande o suficiente e bem distribuído para que seja bem representativo fazendo com que o risco empírico possa se aproximar ao risco funcional. O conjunto de dados de treinamento geralmente não possui as características necessárias para que a minimização do risco empírico seja altamente consistente, o que conduz a problemas quanto à capacidade de generalização, devido ao fato de que o processo de treinamento poderá gerar soluções que se aproximam dos dados de treinamento, modelando o ruído presente nos dados e não representando bem a função geradora. Este super-ajuste aos dados de treinamento é denominado *overfitting*. Para que este problema seja minimizado os parâmetros da rede devem ser muito bem ajustados além de se ter a necessidade de definir uma topologia adequada. A definição da topologia da rede do tipo perceptron de múltiplas camadas (do termo *multi layer perceptron* - MLP em Inglês), que é utilizada neste trabalho, trata-se basicamente da definição da quantidade de camadas escondidas da rede e da quantidade de neurônios em cada camada. Quanto maior a quantidade de neurônios na rede, maior é o espaço para busca de soluções. Com um espaço de alta dimensionalidade, as soluções encontradas após o treinamento podem sofrer grande variabilidade considerando-se diferentes execuções do algoritmo para um mesmo conjunto de treinamento, o que é definido como variância [15].

A minimização da variância torna-se necessária para que sejam gerados modelos com boa capacidade de generalização, minimizando o problema do super-ajuste aos dados de treinamento. Por outro lado uma minimização excessiva da variância conduz a um sub-ajuste(*underfitting*) aos padrões de treinamento, fazendo com os modelos obtidos não representem bem o problema [38]. Desta maneira a busca por uma maior capacidade de generalização conduz ao ajuste entre a complexidade dos modelos gerados e a complexidade do problema, buscando encontrar um modelo que não seja demasiadamente rígido a ponto de não modelar os dados e nem flexível demais a ponto de modelar o ruído [5].

Em resumo no treinamento de redes neurais tem-se a necessidade de minimizar o risco e controlar a complexidade da rede para que se tenha um maior poder de generalização. Os algoritmos clássicos para o treinamento de rede visam apenas à minimização do risco empírico, geralmente através da minimização do erro médio quadrático. Recentemente têm surgido diversos algo-

ritmos que visam controlar risco e complexidade através de um treinamento multi-objetivo.

A complexidade da rede neural pode ser dividida de duas formas: complexidade estrutural e complexidade aparente. O controle da complexidade estrutural se resume em manipular diretamente a estrutura da rede, alterando as conexões e quantidade de neurônios em suas camadas. A complexidade aparente tem foco voltado para a magnitude dos pesos da rede, não levando em consideração a estrutura física. Em [15] é mostrado que a variância é proporcional à magnitude dos pesos e em [2] é visto que a magnitude dos pesos é mais importante do que a quantidade deles na rede quando se busca uma rede neural como maior poder de generalização. A seleção de modelos visando maior capacidade de generalização tem sido obtida através de técnicas como a utilização de métodos de validação, algoritmos construtivos, algoritmos de poda e regularização. Nas abordagens por validação uma medida de erro sobre um conjunto de validação [35] é usada para selecionar os modelos sem, no entanto, fazer referências explícitas à complexidade dos mesmos. Algoritmos de poda ou construtivos [30] são baseados na manipulação da estrutura (ou topologia) da rede que, na prática, resultam em modelos menos complexos apresentando um número menor de parâmetros. Aplicações de técnicas de regularização, como o conhecido método *Weight Decay* (WD) [20], são geralmente baseadas na ideia de controlar a complexidade dos modelos a partir de restrições na magnitude dos pesos da rede. Nessa mesma linha, novos algoritmos de aprendizado baseados em técnicas de otimização multi-objetivo (MOBJ) têm sido aplicados ao aprendizado de RNAs [29] [8] [22]. Estes métodos visam a seleção de modelos através da minimização de dois objetivos conflitantes: o erro de treinamento e a norma do vetor de pesos. Após o processo de otimização, um conjunto de soluções de eficientes (não dominadas) constituindo uma estimativa do conjunto Pareto-Ótimo [21] é obtida. Dentre as soluções obtidas, aquela que apresenta complexidade compatível com a tarefa de aprendizagem em questão deve ser selecionada.

Em problemas com grande dimensionalidade, como problemas com bases de dados de genes, que possuem milhares de variáveis, o treinamento de RNAs torna-se ainda mais complexo devido ao fato de se ter necessariamente uma topologia de maior extensão, gerada pelo grande volume de dados presente no problema, o que conduz a modelos com alta variância. Além do alto tempo computacional necessário para o treinamento de uma RNA utilizando este tipo de base de dados, o aprendizado da rede pode sofrer da 'maldição da dimensionalidade' [14] devido ao pequeno número de amostras encontrado neste tipo de problema. Neste sentido é recomendável a redução de dimensão

do problema para viabilizar a utilização deste em aprendizado de máquina. É possível se obter um conjunto reduzido de características que represente bem o problema através da seleção de características mais relevantes e eliminação de redundâncias.

Verificando-se as considerações descritas, propõe-se neste trabalho dois métodos para o treinamento multi-objetivo de RNAs e uma nova abordagem para seleção multi-objetivo de características. Os novos métodos para treinamento de rede utilizam a técnica ϵ-restrito vista no MOBJ como em [21] e [29], mas eliminam a necessidade de utilização de um algoritmo para otimização não-linear com tratamento de restrições. O tratamento de restrições torna-se desnecessário durante o processo de treinamento da rede, dado que estas não são tratadas de forma explícita na função objetivo definida. A abordagem para seleção de características é testada aqui para o problema de classificação de tipos de leucemia aguda e tem o objetivo de buscar o menor conjunto de características que maximize a taxa de classificações corretas.

1.1 Objetivos: Geral e Específicos

A Motivação deste trabalho está no fato de se explorar novas características do treinamento de redes neurais que conduzam à obtenção de soluções que maximizem a capacidade de generalização sem tornar complexo o processo de treinamento da rede. Desta maneira, a redução de problemas com alta dimensionalidade é interessante na implementação de classificadores mais eficientes.

Neste contexto, o objetivo deste trabalho é o desenvolvimento de algoritmos para o treinamento de RNAs que conduzam a melhorias quanto à capacidade de generalização. Isto é feito através da geração de uma estimativa do conjunto pareto ótimo de forma que se possa manter a simplicidade do problema de otimização da rede, efetuando um treinamento mono-objetivo sem o tratamento de restrições. Objetiva-se também o desenvolvimento de uma abordagem para seleção de características visando a obtenção de um conjunto reduzido de atributos que maximize a taxa de classificação correta, considerando-se no processo, uma avaliação multivariada das características para determinação da relevância destas em conjunto.

Os objetivos específicos são:

1. Desenvolvimento de algoritmos para o treinamento multi-objetivo de RNAs sem a necessidade de se tratar restrições no processo de otimização.

2. Avaliação dos métodos propostos aplicados a problemas de regressão e classificação.

3. Desenvolvimento de uma abordagem para seleção de características com avaliação multivariadas que obtenha um conjunto reduzido de características com boa taxa de classificação.

1.2 Contribuições

O primeiro método para treinamento de rede utiliza o algoritmo Evolução Diferencial (em inglês, *Differential Evolution* - DE) para o treinamento de redes MLP, sendo denominado DEANN. Define-se o método de forma que os pesos da rede tem magnitude limitada através de valores fixos de norma estabelecidos, sendo esta restrição acoplada à codificação utilizada no algoritmo. Desta maneira existe a necessidade de se minimizar apenas o erro médio quadrático de forma direta. Comparações deste método em relação a algoritmos clássicos para o treinamento de RNAs podem ser vistos em [31].

O outro método proposto para o treinamento de rede também visa o controle da complexidade aparente da rede através da minimização de erro e norma. Nesta abordagem um novo espaço para busca de soluções é gerado em decorrência da introdução de uma nova formulação para representação do problema de otimização da rede. Baseada na função norma dos pesos da rede a formulação utiliza conceitos de coordenadas esféricas para modelar o problema de otimização, de forma que um dos objetivos é tratado de forma intrínseca, onde dado um valor de norma o treinamento busca pelo menor erro no espaço limitado por esta norma nas condições da nova formulação. Este método é denominado Busca pela Superfície da Norma (em inglês, *Norm Surface Search* - NSS).

A Figura 1.1 mostra a limitação imposta à superfície de busca a ser percorrida pelo método de otimização utilizando a formulação no sistema de coordenadas esférico. Pode ser visto que dado um valor de norma, apenas um pequeno espaço da região total pode ser percorrido o que faz com que a única preocupação no processo de otimização seja encontrar o menor erro neste espaço limitado, que pode ser melhor visualizado na Figura 1.2 em que pode-se notar a solução de menor erro para um determinado valor da norma.

Os resultados encontrados por ambos os métodos propostos foram promissores. Verificando-se a média das métricas utilizadas, nos problemas de regressão os erros obtidos tiveram diferenças muito pequenas em relação ao MOBJ e, em relação ao tempo computacional, o DEANN foi equivalente ao

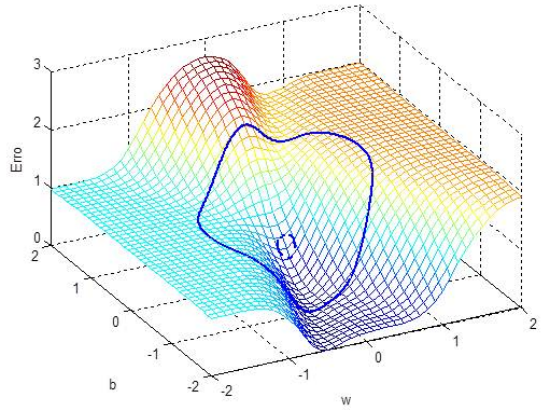

Figura 1.1: Curvas de erro para valores constantes de norma.

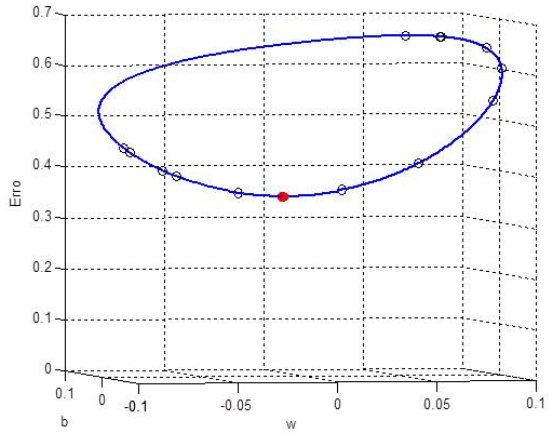

Figura 1.2: Solução no espaço limitado por um determinado valor de norma.

MOBJ enquanto que o NSS teve um alto tempo computacional. Em relação aos problemas de classificação os dois métodos propostos foram superiores ao MOBJ no quesito acurácia, exceto para a base de maior dimensão em que o MOBJ e o NSS foram equivalentes sendo superiores ao DEANN. Em relação ao tempo computacional para os problemas de classificação verificou-se os mesmos resultados encontrados nos problemas de regressão.

Em relação à abordagem para seleção de características, é denominada híbrida, por utilizar filtros uni-variados para pré-seleção e um método multivariado, a seleção clonal, para seleção final das características. Após a realização da seleção utilizou-se dois classificadores muito explorados na literatura para se avaliar os resultados da abordagem, verificando-se bons resultados com a utilização do método proposto.

1.3 Organização da dissertação

O capítulo 2 traz um pequeno resumo sobre os conhecimentos necessários ao entendimento do que será tratado ao longo do trabalho com várias referências a trabalhos que abordam os temas mais a fundo.

O capítulo 3 detalha o treinamento multi-objetivo de redes neurais utilizando o algoritmo Evolução Diferencial e mostra a abordagem utilizada para seleção multi-objetivo de características.

O capítulo 4 explica com detalhes a formulação baseada em coordenadas hiperesféricas para o treinamento de redes neurais artificiais.

O capítulo 5 detalha a metodologia utilizada para aplicação e realização de testes dos métodos e abordagens propostos neste trabalho.

O capítulo 6 apresenta os resultados obtidos após os testes e simulações realizadas com os métodos para treinamento multi-objetivo de redes neurais e com a abordagem para seleção multi-objetivo de características.

O capítulo 7 apresenta as conclusões e perspectivas futuras para o trabalho.

CAPÍTULO 2

Revisão Bibliográfica

Neste capítulo pretende-se discorrer brevemente sobre os assuntos necessários para a compreensão do trabalho desenvolvido. São os conceitos básicos e as definições envolvidas, assim como a literatura utilizada para pesquisa e que também são fonte de explicações mais detalhadas.

2.1 Treinamento de Redes Neurais Artificiais

Redes neurais artificiais (RNAs) podem ser definidas como modelos computacionais que têm por objetivo a utilização de fatores conhecidos sobre o funcionamento do cérebro humano para a obtenção de melhores resultados na resolução de problemas em que métodos computacionais tradicionais têm se mostrado inadequados.

Para [5] as RNAs são sistemas paralelos distribuídos compostos por unidades de processamento simples (neurônio artificial), que calculam determinadas funções matemáticas (normalmente não-lineares). Elas podem ser de uma ou múltiplas camadas e estarem interligadas por várias conexões, geralmente unidirecionais.

De acordo com [19] uma rede neural é similar ao cérebro humano em dois aspectos:

1. O conhecimento é obtido pela rede de seu ambiente, por meio de um processo de aprendizado.

2. O armazenamento do conhecimento adquirido, que acontece através da

utilização das forças de conexões entre os neurônios, conhecidas como pesos sinápticos.

O processo de aprendizagem em uma rede neural é denominado algoritmo de aprendizagem. Sua função é alterar os valores dos pesos sinápticos, de forma ordenada, visando o alcance de um objetivo de projeto desejado. Este é o método tradicional para projetos de redes neurais.

Maiores detalhes sobre os modelos de neurônios e arquiteturas de redes neurais podem ser encontrados em [5] e [19].

2.1.1 Redes MLP

Redes de perceptrons de múltiplas camadas consistem num importante modelo de rede neural artificial que tem sido aplicado com sucesso em diversos tipos de problemas difíceis, com um treinamento supervisionado geralmente através do popular algoritmo de treinamento conhecido como algoritmo de retro propagação de erro (*back-propagation*) e suas variações.

Neste modelo de rede utilizam-se camadas intermediárias que introduzem não linearidade entre a entrada e a saída da rede. Desta maneira a rede ganha maior capacidade para resolução de problemas complexos devido à maior extração de conhecimento e aumento do poder discriminativo proporcionado pela projeção dos padrões de entrada da rede num espaço não linear de alta dimensão.

Como pode ser visto em [5] algoritmos de minimização do erro utilizados para treinamento de redes neurais, como o *back-propagation*, geralmente utilizam informações sobre a superfície local do erro para obtenção de melhorias na atualização dos pesos. O gradiente descendente, método geralmente utilizado nestes algoritmos, gera uma aproximação linear da função de erro em relação aos pesos. Este algoritmo tem como principal vantagem a facilidade de implementação, mas com desempenho altamente influenciado pela superfície de erro gerada pelo problema em questão.

Como mostrado em [8] o desempenho de generalização é a principal preocupação no treinamento de redes neurais. O *back-propagation* e suas variações, em suas definições originais, não levam em consideração esta dificuldade, dado que por levar em consideração somente o erro do modelo gerado tem se garantias de aproximação apenas para os dados usados no treinamento da rede que em geral são amostrados de uma distribuição desconhecida.

Em [5] pode ser visto que diversos fatores influenciam o treinamento de redes MLP, tais como, as definições da quantidade de neurônios nas camadas ocultas, taxa de aprendizagem, heurísticas para inicialização dos pesos, etc.

Ao fim do treinamento busca-se um modelo que apresenta boa aproximação para os dados de treinamento e também, boa capacidade de generalização. O ajuste da complexidade do modelo junto à minimização do erro busca encontrar um modelo que não seja demasiadamente rígido a ponto de não modelar os dados e nem flexível demais a ponto de modelar o ruído. Este ajuste entre o erro de treinamento e a complexidade do modelo é conhecido como dilema entre o viés e a variância [15]. Considerando-se uma rede MLP, conhecendo-se a relação entre a capacidade de aprendizagem e o tamanho dos pesos da rede, pode-se definir a norma dos pesos da rede $\|w\|$ como uma medida de controle da variância [2]. Desta maneira pode-se definir um modelo que controle erro e a complexidade em termos de um modelo de otimização mono-objetivo com restrições, descrito como:

$$\begin{array}{c} \min_{w \in W} J(w) = \frac{1}{N} \sum_{i=1}^{N} (d_i - y_i)^2 \\ sujeito a: \\ \|w\| \leq \epsilon \end{array} \tag{2.1}$$

onde d_i e y_i são os valores para a saída desejada e a saída da rede para i-ésima amostra, respectivamente, sendo que, $J(w)$ é o erro quadrático médio e ϵ é um valor arbitrário que limita a norma dos pesos da rede.

2.1.2 Otimização Multi-objetivo

Os problemas de otimização multi-objetivo consistem na busca pelo conjunto de variáveis que gere um conjunto viável de soluções que satisfaçam algumas restrições e otimize um vetor de funções que representa uma função objetivo. Desta maneira, objetiva-se encontrar uma solução do conjunto obtido em que os valores de todas as funções são considerados aceitáveis pelo projetista [27]. A formulação matemática para problemas desta natureza é detalhada nos trabalhos de [34], [13] e [23].

O conjunto Pareto-Ótimo

Dado um problema de otimização multi-objetivo, tem-se a necessidade de encontrar um conjunto em que as soluções são aceitáveis, dado um conjunto de restrições e funções objetivo. Assim sendo, o conceito de eficiência [28] faz-se necessário, consistindo em dizer que uma determinada solução é considerada ótima se não existe nenhuma outra solução viável que melhore um objetivo sem piorar pelo menos um dos outros presentes no vetor de objetivos. Desta forma esta solução é dita pertencente ao conjunto Pareto-Ótimo, modificando a definição de ótimo, dado que agora não se trata de apenas uma

solução, mas todas as soluções que satisfaçam as condições para serem consideradas eficientes, o que dá origem a um conjunto de soluções denominado conjunto Pareto-Ótimo.

2.2 Métodos Evolucionários para Busca e Otimização

Os métodos baseados em computação evolucionária utilizam conceitos de sistemas biológicos como inspiração para criação de ferramentas computacionais para resolução de problemas de busca. Estes métodos estão contidos na classe de algoritmos estocásticos, sendo que a busca é baseada em regras probabilísticas. Características fundamentais para a escolha destes métodos na resolução de problemas são a eficácia na busca por ótimo global na superfície de funções e a não necessidade de se utilizar derivadas. Esta seção mostra de forma sucinta os aspectos fundamentais dos métodos evolucionários utilizados neste trabalho: Algoritmo de Seleção Clonal (CLONALG) e Evolução Diferencial (*Differential Evolution - DE*). O CLONALG foi utilizado como algoritmo de busca num processo de seleção de características enquanto que o DE foi utilizado para otimização de RNAs.

2.2.1 Sistemas Imunológicos Artificiais

O sistema imunológico é responsável pela principal forma de proteção do hospedeiro contra agentes infecciosos. Podem ser geradas duas formas de resposta a estes invasores, uma rápida e efetiva efetuada pelo sistema imune inato e outra mais lenta e duradoura oriunda do sistema imune adaptativo [9]. As células do sistema imune inato constituem uma resposta a diversos patógenos invasores sem a exigência de uma exposição anterior a estes enquanto o sistema imune adaptativo gera uma resposta imune específica a um determinado agente infeccioso com produção de anticorpos para este patógeno. Qualquer molécula reconhecida pelo sistema imunológico adaptativo é denominada antígeno (Ag). A geração de anticorpos (Abs) é feita pelos linfócitos B (ou células B). Estas células são capazes de desenvolver uma memória imunológica que permite a identificação de um estímulo antigênico caso este seja novamente exposto ao sistema imune, evitando assim uma possível nova infecção. Os sistemas imunológicos artificiais inspiram-se nas definições acima citadas e, através das características básicas do sistema imune biológico, constroem ferramentas computacionais que auxiliam na resolução de complexos problemas de engenharia.

2.2.2 Algoritmo de Seleção Clonal

Na imunologia o princípio da seleção clonal define que células que reconhecem antígenos são selecionadas para proliferar, passando por um processo de clonagem através de sucessivas mitoses. Estes clones estão sujeitos a mutações somáticas a altas taxas e uma força seletiva formando o processo de maturação de afinidade, onde os níveis de afinidade das células são melhorados em relação aos antígenos reconhecidos. Outro mecanismo a ser considerado é a edição de receptores, onde células com baixo nível de afinidade são substituídas por células totalmente novas, visando manter a diversidade populacional [10]. Baseado nestes conceitos o algoritmo de seleção clonal utiliza conceitos básicos do funcionamento do sistema imunológico biológico para formulação de ferramentas para resolução de diversos problemas complexos de engenharia. O algoritmo de seleção clonal pode ser também considerado um algoritmo evolucionário, devido às características de diversidade, variações genéticas e seleção natural presentes nele. O algoritmo CLONALG proposto em [11] demonstra uma aplicação computacional dos princípios de seleção clonal e maturação de afinidade aplicada inicialmente a tarefas de aprendizagem de máquina e reconhecimento de padrões, sendo posteriormente adaptada a problemas de otimização. O algoritmo implementado neste trabalho tem por objetivo a resolução de problemas de otimização, utiliza uma representação binária para os Abs e consiste dos seguintes passos:

1. Geração de uma população inicial aleatória de Abs, denominada conjunto Ab.

2. Avaliação da afinidade dos indivíduos presentes em Ab em relação à função objetivo.

3. Seleção dos $b\%$ Abs com maior afinidade em Ab, compondo uma subpopulação Abn.

4. Clonar os Abs presentes em Abn, formando um conjunto de clones C, sendo o número de clones de cada Ab proporcional à afinidade dos mesmos, onde Abs com maiores afinidades possuem um maior número de clones.

5. Submissão do conjunto de clones C ao processo de maturação de afinidade, onde sofrem mutações em altas taxas, inversamente proporcionais aos seus níveis de afinidade. Abs com maiores afinidades têm menores taxas de mutação. Ao fim deste processo é gerado um conjunto Cm de clones maturados.

6. Avaliação dos Abs do conjunto Cm de clones maturados.

7. Seleção dos Abs do conjunto Cm com maiores afinidades para compor a população Ab. Um determinado Ab presente em Cm que tenha afinidade maior que seu respectivo representante na população Ab substitui este.

8. Substituir os $d\%$ Abs com menores afinidades em Ab por novos indivíduos gerados aleatoriamente.

Esta sequência de passos se repete a partir do passo 2 até que se alcance um critério de convergência para o algoritmo. Após selecionar-se os $b\%$ Abs com maiores afinidades da população Ab (passo 3) o processo de clonagem (passo 4) é regido por:

$$N_c = \sum_{i=1}^{n} round\left(\frac{\beta * N}{i}\right) \tag{2.2}$$

onde N_c é o número de total de clones gerados na etapa de clonagem, β é um fator de multiplicação, N é o total de Abs da população Ab e $round()$ é utilizado para arredondar o valor da função para o inteiro mais próximo. Cada parcela do somatório presente na função representa a quantidade de clones de um elemento Ab_i sendo que estes elementos estão ordenados de forma decrescente em relação à afinidade, onde i representa o índice destes elementos ordenados. No processo de maturação de afinidade a taxa de mutação é proporcional à afinidade dos indivíduos. Em [11] o cálculo da taxa de mutação é dado por:

$$\alpha = exp(-\rho * f) \tag{2.3}$$

onde α é o tamanho do passo, ρ é o fator que controla o decaimento da função e f é a afinidade do indivíduo normalizada no intervalo [0;1]. A utilização de limites mínimos e máximos para a taxa de mutação pode auxiliar numa busca mais eficiente. A relação entre a afinidade dos indivíduos e a taxa de mutação pode ser visualizada na Figura 2.1, onde nota-se claramente a forte influência do parâmetro no desempenho do algoritmo. A operação de mutação no algoritmo de seleção clonal consiste na operação chave para determinar o desempenho do algoritmo de otimização em relação à velocidade de convergência e eficácia na busca pela solução ótima [26]. A mutação proporcional à afinidade representa um processo de busca local na superfície da função. O processo de edição de receptores, representado pela substituição dos indivíduos com menores afinidades da população Ab por novos indivíduos gerados aleatoriamente, impõe a introdução e manutenção da diversidade populacional, efetuando uma busca global através da exploração de novas regiões na superfície total de busca. Características como a aplicação a problemas de

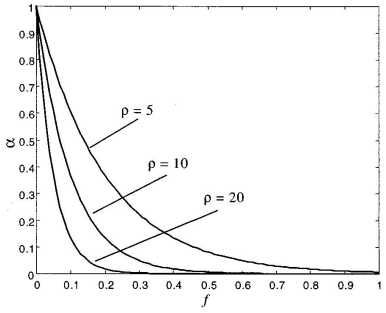

Figura 2.1: Relação entre a taxa de mutação e a afinidade do indivíduo.

aprendizagem de máquina e otimização, busca por vários ótimos em funções multimodais e um pequeno número de parâmetros para se ajustar fazem com que o Clonal torne-se recomendável à aplicação em diversos problemas em diversas áreas de pesquisa.

2.2.3 Algoritmo Evolução Diferencial

O algoritmo evolução diferencial se enquadra na classe de algoritmos evolutivos, mesmo não possuindo uma inspiração em sistemas naturais. A mutação diferencial, operação que define o algoritmo, possuí fundamento baseado em princípios matemáticos e heurísticos, mas por utilizar uma população de soluções que evoluem durante as iterações do algoritmo, além de operadores heurísticos relacionados a mecanismos gerais de adaptação natural, este possui características encontradas nos algoritmos evolutivos de forma geral. Proposto em [36] o algoritmo Evolução Diferencial foi desenvolvido para resolver problemas de otimização, mas no intuito de ser uma técnica com rápida convergência e de fácil utilização, utilizando poucos parâmetros a serem definidos pelo usuário. Trata-se num algoritmo simples e eficiente que têm se destacado no âmbito da otimização não linear com variáveis continuas. A ideia fundamental que sustenta o algoritmo é o esquema de mutação diferencial onde se gera um "vetor tentativa"(ou vetor diferencial), obtido através da diferença vetorial ponderada entre dois indivíduos da população, adicionada a um terceiro indivíduo. Este vetor diferencial gerado é comparado a um indivíduo pré-determinado da população e substitui este caso possua um valor mais adequado à função objetivo, dada a tarefa de otimização em questão, se minimização ou maximização. É mostrado em [17] que durante o processo de busca do algoritmo os tamanhos e orientações dos vetores diferenciais são modificados de acordo com a paisagem da função o objetivo de forma a se ajustarem a ela, conferindo desta forma uma característica de auto adaptação ao DE. Com esta característica a mutação diferencial provê ao algoritmo maior

robustez, versatilidade e eficiência para a resolução de diversos problemas. A estrutura básica do DE é semelhante à maioria dos algoritmos evolutivos, como pode ser visto com detalhes em [24], consistindo dos passos a seguir na implementação utilizada:

1. Inicialização da população inicial de forma aleatória dentro dos limites dos parâmetros.

2. Avaliação dos indivíduos da população quanto sua adequação à função objetivo.

3. Para cada indivíduo w_i da população efetua-se o cruzamento do individuo com o vetor diferencial (v) gerado com a operação mutação diferencial, obtendo-se um indivíduo u_i.

4. Seleção da população para a nova iteração mantendo o indivíduo mais adequado à função objetivo tomando-se a comparação entre μ_i e w_i na iteração atual.

Esta sequência de passos se repete a partir do passo 2 até que se alcance um critério de convergência para o algoritmo. A inicialização da população inicial ocorre após definir-se os valores de limite inferior (w_{min}) e superior (w_{max}) para os parâmetros utilizados, em seguida, gera-se indivíduos distribuídos uniformemente dentro desta faixa determinada, sendo definido por:

$$w_i(w) = w_{min} + rand * (w_{max} - w_{min}) \tag{2.4}$$

onde $rand$ é um valor aleatório no intervalo $[0; 1]$ obtido a partir de uma distribuição uniforme. Todas as variáveis de um indivíduo w_i possuem seus valores dentro da mesma faixa. Após definir-se a população inicial percorre-se esta passando por cada indivíduo efetuando-se uma operação de cruzamento (ou recombinação discreta) do individuo atual no processo com um vetor diferencial gerado pelo processo de mutação diferencial. Este vetor é obtido, em uma versão original do DE, por:

$$v = w_{r1} + \eta(w_{r2} - w_{r3}) \tag{2.5}$$

onde w_{r1}, w_{r2} e w_{r3} são vetores obtidos aleatoriamente da população, sendo que estes são obtidos de forma mutuamente excludentes. w_{r1} é denominado vetor base e η é um fator de escala aplicado ao vetor diferença. A operação de

recombinação discreta é definida por:

$$\mu_i^j = \left\{ \begin{matrix} v^j, \ se \ rand \leq C \ ou \ j{=}\delta \\ w_i^j, \ caso \ contrrio \end{matrix} \right\} \tag{2.6}$$

onde $j = 1, ..., n$ é o índice de uma determinada variável do individuo, C é a probabilidade de cruzamento, definido no intervalo $[0; 1]$ e, $\delta \in \{1, ..., n\}$ é um índice aleatório definido para que pelo menos uma variável do indivíduo herde características da solução mutante.

2.3 Seleção de Características

Em problemas com grande dimensionalidade, como é o caso dos problemas com expressões gênicas, existem muitos atributos irrelevantes e um número reduzido de amostras, o que ocasiona em aumento de complexidade computacional e perda de exatidão na tarefa de classificação. Nestes casos torna-se necessário a remoção de características irrelevantes e a definição de um subconjunto reduzido de características discriminativas para melhorias na classificação [37]. Uma desvantagem da seleção de características é o aumento de uma camada de complexidade no processo devido ao custo de se obter um subconjunto adequado à resolução do problema num espaço de busca relativamente grande. No contexto de classificação as técnicas de seleção de características se diferem quanto à forma utilizada para incorporar a busca no espaço adicional dos subconjuntos de características à escolha do modelo, dividindo-se em três categorias: métodos de filtro, métodos *wrapper* e métodos embarcados [33]. Os métodos de filtro e *wrapper* se diferem na forma de avaliação dos subconjuntos de características. Os filtros utilizam critérios baseados em informações intrínsecas aos dados sem utilização de nenhuma técnica de aprendizagem de máquina enquanto que *wrappers* utilizam o desempenho de uma máquina de aprendizagem treinada utilizando um subconjunto específico de características. Os métodos de filtros são também conhecidos como métodos de rankeamento de características, pois na maioria dos casos realizam o cálculo de um índice de relevância das características em relação à discriminação obtida em relação às categorias encontradas nos dados. Filtros e *wrappers* também podem ser combinados formando técnicas híbridas onde se utiliza os filtros para criação do rank e, em seguida, utiliza-se uma abordagem *wrapper* levando em consideração as características mais relevantes. Estas duas técnicas utilizam estratégias de busca para explorar o espaço de subconjuntos de características devido à inviabilidade

de se efetuar uma busca exaustiva num espaço com muitas dimensões. Nos métodos embarcados a busca por um subconjunto ótimo de características é realizada dentro do processo de construção da máquina de aprendizagem [18]. Neste trabalho utilizou-se um método híbrido para seleção de características. Os filtros utilizados são baseados em análise uni-variada, onde se realiza uma análise relativa à relevância individual de cada uma das características considerando-se independência entre elas. Estes filtros são vistos a seguir enquanto a estratégia de busca utilizada e a abordagem *wrapper* serão vistos nas seções posteriores.

2.3.1 F-Score

O filtro *F-Score* (*Fisher score*) é um critério simples e eficiente que, através de características estatísticas dos dados, mede a relevância das características para discriminação entre classes [6]. Considerando-se o um problema de classificação binário com as classes $C1$ e $C2$, ele é definido por:

$$f(i) = \frac{(\mu_i^{c_1} - \mu_i) + (\mu_i^{c_2} - \mu_i)}{\sigma_i^{c_1} + \sigma_i^{c_2}} \tag{2.7}$$

onde i corresponde ao índice da i-ésima características e, μ_i^c e σ_i^c são média e desvio padrão para a classe C em relação à característica i.

2.3.2 Pearson Correlation Coeficient

O coeficiente de correlação de Pearson é outro método geralmente utilizado para rankear características em relação ao seu poder discriminativo para as classes em problemas de classificação binários [18], sendo definido por:

$$f(i) = \frac{\sum_{i=1}^{p} (x_{ij} - \bar{x})(y_i - \bar{y})}{\sigma_{xj}\sigma_y} \tag{2.8}$$

onde j corresponde à j-ésima característica, i é um padrão de entrada e p é o número total de amostras. O vetor xj contem todos os valores da característica j para todas as amostras de treinamento e y é o vetor contendo todos os valores alvos representando a classe referente a cada amostra.

2.4 Classificação de Padrões

A classificação de padrões é a tarefa em que se atribui um determinado objeto (padrão) a uma categoria (classe), dado um conjunto de características (também chamado conjunto de variáveis ou atributos) que representam este

objeto. Nesta tarefa, de forma geral, determina-se a probabilidade de um objeto pertencer a uma determinada categoria, sendo geralmente impossível uma classificação ótima [12]. O classificador de Bayes e o classificador baseado na regra dos k vizinhos mais próximos (em inglês, *k-nearest-neighbor - K-NN*), são técnicas geralmente utilizadas na tarefa de classificação de padrões.

2.4.1 Classificador de Bayes

O Classificador de Bayes baseia-se na suposição de que o problema de decisão é visto de uma forma probabilística onde se conhece todos os valores de probabilidades relevantes. A classificação de um objeto a uma determinada classe é feita de acordo com a probabilidade de o objeto pertencer à classe [12]. Um classificador de Bayes simples (também denominado classificador de Bayes ingênuo) supõe independência entre as variáveis, o que não ocorre na maioria dos problemas de classificação, mas ainda assim obtém resultados competitivos com a maioria dos classificadores além de possuir menor complexidade computacional devido à facilitação nos cálculos utilizados obtida pela suposição de independência. A fórmula geral utilizada pelo classificador de Bayes é dada por:

$$P(C_j|X) = \frac{P(X|C_j)P(C_j)}{P(X)} \tag{2.9}$$

onde $P(C_j|X)$ é o termo definido como probabilidade à posteriori que indica a probabilidade da classe ser C_j dado que o padrão X foi mensurado. O termo $p(X|C_j)$ é uma probabilidade condicional denominada verossimilhança que representa a probabilidade de X dado que a classe C_j foi apresentada e $P(C_j)$ é a probabilidade a priori, sendo a informação que reflete o conhecimento prévio que se tem sobre os dados em relação à predição de determinado objeto pertencer à classe C_j levando em consideração apenas as quantidades de objetos amostrados em cada classe. O termo $p(X)$ é definido como evidência e pode ser visto como um mero fator de escala que garante que a soma das probabilidades à posteriori é igual a um.

2.4.2 K-NN

O *K-NN* é um classificador também muito utilizado na literatura, como em [7], e pertencente à categoria dos algoritmos de aprendizagem baseados em memória. Neste classificador os dados de treinamento são utilizados para formação de uma memória de exemplos com padrões de entrada e suas respectivas saídas corretas. Neste contexto, a classificação de um padrão ainda desconhecido ocorre através da análise dos padrões armazenados na memó-

ria, onde se atribui o rótulo de determinada classe a este padrão de acordo com a classe dos k padrões mais similares a ele, levando-se em consideração alguma métrica de distância para avaliação da similaridade [19]. Uma medida geralmente utilizada para se avaliar a similaridade entre padrões é a distância euclidiana.

2.5 Considerações finais

Neste capítulo deu-se uma visão geral sobre os assuntos, definições e conceitos básicos que são relevantes ao entendimento do trabalho desenvolvido e que será explicado nos próximos capítulos.

CAPÍTULO
3

Utilizando Algoritmos Evolucionários para o Treinamento Multi-Objetivo de RNAs e Seleção de Características

Neste capítulo é detalhado o processo de treinamento de uma RNA através do algoritmo Evolução Diferencial. É mostrado também neste capítulo, uma abordagem para seleção multi-objetivo de características com busca realizada através do algoritmo de Seleção Clonal.

3.1 Treinamento Multi-Objetivo de RNAs Utilizando o Algoritmo Evolução Diferencial

O método proposto, aqui denominado DEANN, utiliza o algoritmo DE para modificação dos pesos da rede de forma a minimizar erro e norma, semelhante à forma que ocorre em [29], substituindo-se o algoritmo de otimização determinístico utilizado. Utilizou-se o erro médio quadrático como medida de erro para efeito de comparação com outros métodos geralmente utilizados na literatura. A medida de fitness utilizada para avaliação das soluções encontradas é o valor do erro médio quadrático encontrado após submeter-se todos os padrões de treinamento à rede. Visando a melhoria na capacidade de generalização, utiliza-se valores fixos para a norma dos pesos para definição da amplitude destes. Sendo que os valores mínimos e máximos para cada peso

são definidos como:

$$w_{min} = -\frac{\epsilon}{\sqrt{nvar}}$$
$$w_{max} = +\frac{\epsilon}{\sqrt{nvar}}$$

onde $nvar$ é a quantidade de parâmetros livres na rede. A operação de mutação diferencial utilizada nesta implementação é diferente daquela vista na estrutura básica do DE descrita no capítulo 2, sendo que esta forma também pode ser vista em [36], sendo definida por:

$$v = w_i + \lambda(w_{best} - w_i) + \eta(w_{r2} - w_{r3}) \tag{3.1}$$

onde w_{best} é o indivíduo com maior valor de fitness na iteração atual e λ é um fator de escala utilizado para o novo elemento gerado, em substituição ao vetor base definido no capítulo 2. Os indivíduos utilizados no algoritmo, apenas são considerados, caso respeitem os valores limite das variáveis durante as operações de mutação diferencial e cruzamento. O critério de parada utilizado pelo algoritmo é o número de gerações.

3.2 Seleção Multi-Objetivo de Características

A abordagem proposta é baseada na utilização de um método híbrido para seleção de características, onde os filtros uni-variados *F-Score* e *Pearson* são utilizados para rankear o conjunto total de características. Nos ranks gerados as características são ordenadas de forma decrescente quanto ao nível de relevância para discriminação entre classes, sendo que as K-melhores características obtidas a partir de cada filtro serão pré-selecionadas e submetidas a um método *wrapper*. O algoritmo Clonal é utilizado como estratégia para a realização da busca combinatória no método *wrapper* visando a obtenção do menor subconjunto de características com melhor desempenho na classificação.

Cada anticorpo representa um subconjunto de características onde cada característica no subconjunto é representada por um bit de informação, sendo que o valor 0 (zero) indica ausência daquela característica e o valor 1 (um) indica sua presença. Para avaliação da afinidade de um determinado anticorpo utiliza-se a função usada em [37] que busca a otimização de dois objetivos: a maximização da exatidão (taxa de classificações corretas) de um classificador e a minimização do tamanho do subconjunto de características, sendo

resumida numa função a ser maximizada e definida por:

$$f(x) = w * c(x) + (1 - w) * \frac{1}{s(x)} \tag{3.2}$$

onde x é um vetor de características que representa um determinado anticorpo, $c(x)$ é a exatidão de um classificador, $s(x)$ é o tamanho do subconjunto de características e w $\in [0, 1]$ é um parâmetro utilizado para ponderar as duas partes da expressão, sendo que a definição de um valor adequado para este parâmetro conduzirá a um compromisso adequado entre exatidão e tamanho do subconjunto de características. O classificador de Bayes foi o algoritmo de aprendizagem treinado utilizado para avaliação da afinidade de cada subconjunto de características.

A utilização desta abordagem é realizada de maneira que inicialmente aplica-se os filtros *F-score* e *Pearson* para rankear as características presentes na base de dados. A utilização destes filtros numa etapa inicial fundamenta-se no fato de que o foco da abordagem é a seleção de características em bases de dados com mais que centenas de dimensões e um pequeno número de amostras, neste caso em específico, bases de dados oriundas do genoma humano contendo milhares de variáveis. Após a obtenção de um rank das variáveis as K-primeiras características obtidas em cada método foram pré-selecionadas definindo assim o tamanho dos anticorpos, que é igual à quantidade de características utilizadas. No passo seguinte o algoritmo Clonal foi utilizado para se efetuar a busca pelo menor subconjunto presente nestas K características que conduzem à melhor exatidão na classificação. Para o treinamento do classificador de Bayes presente no método *wrapper* utiliza-se a porção dos dados definida como conjunto de treinamento, um conjunto de validação deve ser extraído para definir a exatidão do classificador de Bayes para cada solução apresentada ao método e um subconjunto de testes deve ser utilizado num momento posterior, após a seleção final de características, com o objetivo de testar o desempenho do método proposto.

3.3 Considerações finais

Neste capítulo foram mostradas duas abordagens multi-objetivo utilizando algoritmos evolucionários, uma para o treinamento de RNAs e outra para seleção de características. Foram detalhadas as modificações propostas na definição dos algoritmos e na função objetivo definida na abordagem para seleção de características.

CAPÍTULO 4

Treinamento Multi-Objetivo de Redes Neurais Artificiais com Formulação Baseada em Coordenadas Hiperesféricas

Este capítulo explica a nova formulação para representação do problema de otimização inerente ao treinamento de redes MLP. Será mostrada também a proposta para treinamento de MLPs utilizando a nova formulação, sendo aqui denominado busca pela superfície da norma (em inglês, *Norm Surface Search - NSS*).

4.1 Os pesos da rede representados no sistema de coordenadas hiperesféricas

Sabe-se que a minimização da norma euclidiana dos pesos é uma boa medida para o controle da complexidade da rede. Desta maneira a minimização da Equação 4.1 faz-se necessária no processo de treinamento visando maior poder de generalização.

$$\sqrt{\sum_{i=1}^{n} w_i^2} \tag{4.1}$$

A função norma euclidiana é, em sua essência, uma elipse. Definindo os pesos da rede a partir da função norma é possível representá-los em função

dos ângulos obtidos dentro desta elipse gerada, para um determinado valor de raio (r), que é definido por um valor fixo da norma. Nesta formulação cada peso da rede define um eixo no espaço. Desta maneira, para um determinado valor de norma, pode-se gerar os valores dos pesos em função dos ângulos existentes entre o vetor de pesos de tamanho r e os eixos da elipse, podendo-se assim garantir que variando-se os valores dos ângulos obtêm-se novos valores para os pesos sem que a solução escape da superfície da norma. Para um neurônio simples contendo um valor de bias (w_0) e um peso (w_1) como mostrado na figura 4.1, a disposição destes pesos utilizando a nova formulação pode ser visualizada na Figura 4.2, sendo que esta Figura representa a Equação 4.2, que define a função norma para um problema com duas dimensões.

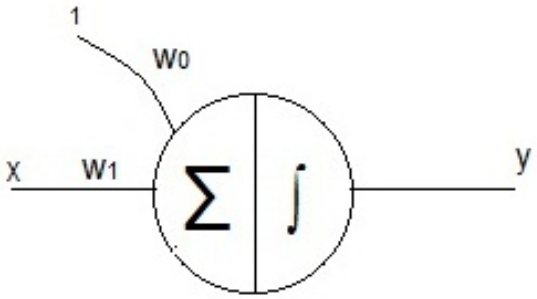

Figura 4.1: Neurônio com uma entrada.

$$r^2 = w_0^2 + w_1^2 \tag{4.2}$$

Os pesos w_0 e w_1 são eixos cuja projeção do raio gera os pesos da rede, r é o valor da norma e θ é o ângulo que, efetivamente, defini os valores dos pesos da rede. Com esta formulação pode-se definir os pesos da seguinte maneira:

$$w_0 = r \sin\theta$$
$$w_1 = r \cos\theta$$

A saída y do neurônio simples com esta formulação pode ser visualizada pela Equação 4.3 em que f é a função de ativação definida para o neurônio.

$$y = f(x * w_1 + w_0) = f(x * r * \cos\theta + r * \sin\theta) \tag{4.3}$$

Com 3 pesos(w_0, w_1, w_2) em um neurônio, existirão 2 ângulos para definir os parâmetros da rede, de forma que deve-se tomar um eixo como referência e projetar os demais em função deste, denotando assim uma representação

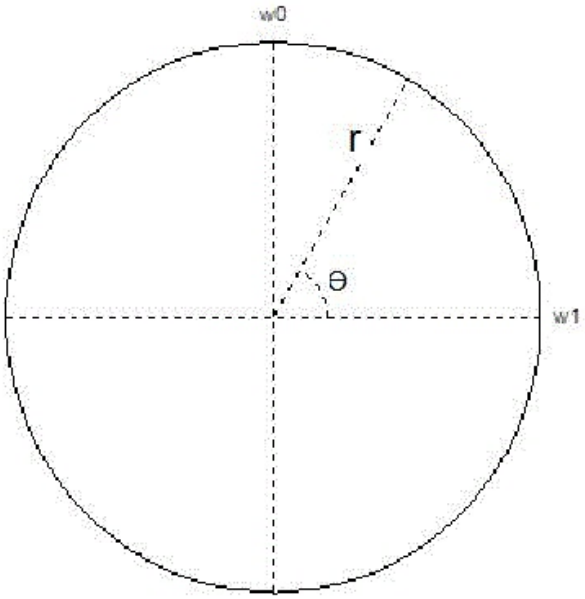

Figura 4.2: Disposição dos pesos na elipse definida pela função norma.

no sistema de coordenadas esféricas [1]. Utilizando o eixo w_0 como referência ter-se-ia a seguinte formulação para os pesos:

$$\begin{aligned} w_0 &= r \sin\theta \\ w_1 &= r \cos\theta \sin\varphi \\ w_2 &= r \cos\theta \cos\varphi \end{aligned}$$

Expandindo-se esta formulação para um problema n-dimensional tem-se a estrutura de uma *n-sphere*(esfera n-dimensional), onde ter-se-ia $n-1$ ângulos representando uma rede com n pesos. Nesta forma de representação do problema de otimização garante-se que os pesos da rede gerem soluções sobre a superfície da norma, além de garantir-se também um limite no domínio das variáveis, onde um dos ângulos terá uma variação na faixa de 0 a 2π e os demais ângulos ficarão entre 0 e π, limitando assim a dimensão do espaço de busca no problema. Pode-se definir esta generalização de representação como sistema de coordenadas hiperesféricas em que os pesos de uma rede com n dimensões podem ser descritos desta forma:

$$\begin{aligned} w_0 &= r \sin\theta_1 \\ w_1 &= r \cos\theta_1 \sin\theta_2 \\ w_2 &= r \cos\theta_1 \cos\theta_2 \sin\theta_3 \end{aligned}$$

$$\vdots$$

$$w_{n-1} = r\cos\theta_1 \cdots \cos\theta_{n-2}\sin\theta_{n-1}$$
$$w_n = r\cos\theta_1 \cdots \cos\theta_{n-2}\cos\theta_{n-1}$$

4.2 Norm Surface Search - NSS

A Figura 4.3 exibe uma superfície de erro para um neurônio com função de ativação não linear. Selecionando-se nesta superfície uma curva de erro em que o valor da norma dos seus parâmetros seja constante, como ilustrado na Figura 4.4 em que são mostradas duas curvas de erro com valor constante de norma, o objetivo do treinamento é encontrar, no espaço limitado por esta curva, o valor de erro mínimo. Por isso a denominação ao método de busca pela superfície da norma (NSS).

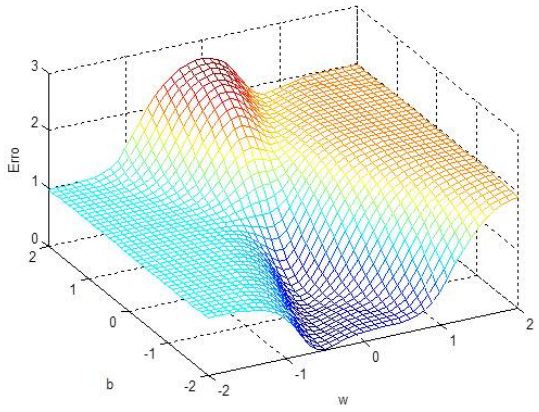

Figura 4.3: Superfície de erro em função dos pesos.

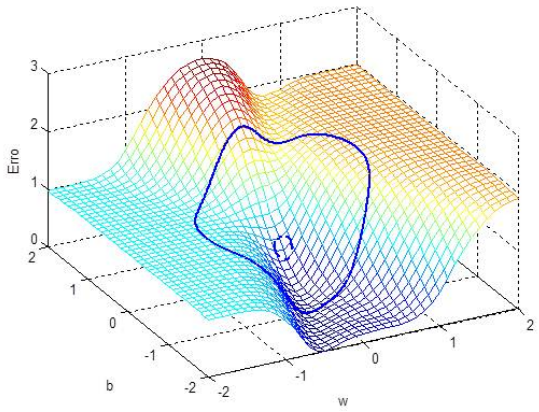

Figura 4.4: Curvas de erro para valores constantes de norma.

Nas figuras 4.3 e 4.4 a visualização da superfície de erro ocorre em função dos pesos da rede. Analisando sob a visão da nova formulação, através dos ângulos que definem os valores dos pesos, tem-se uma superfície com uma dimensão a menos como pode ser visto na Figura 4.5.

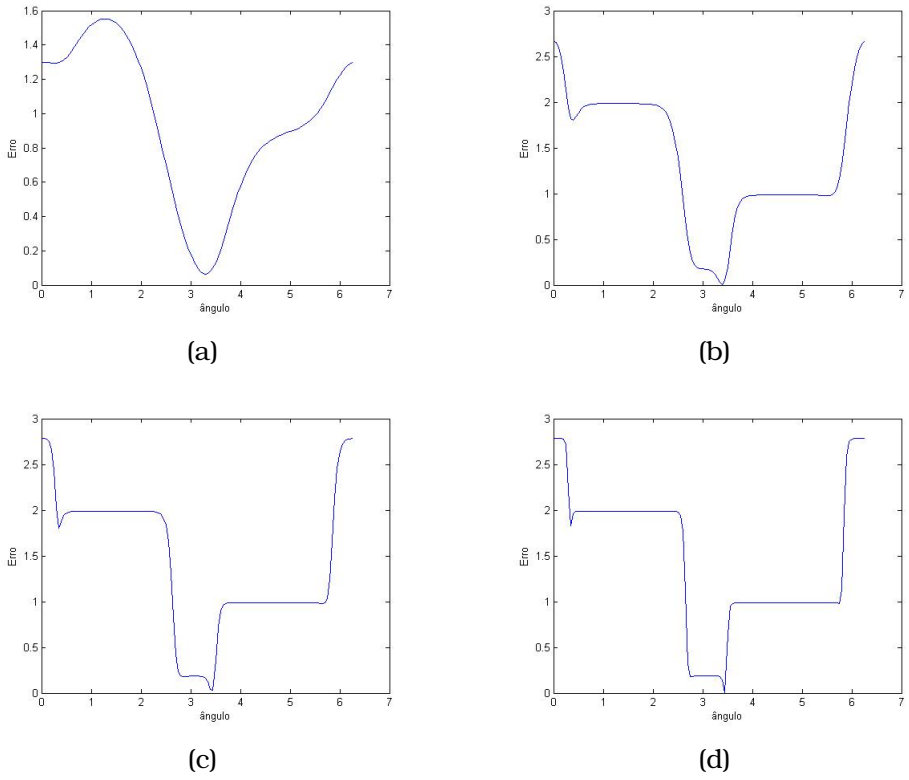

Figura 4.5: Superfície de erro em função dos ângulos. (a) Norma = 0.5, (b) Norma = 2, (c) Norma = 4 e (d) Norma = 8.

Como pode ser visualizado, a superfície de erro torna-se mais complexa à medida que se utiliza maiores valores de norma, em que a superfície inicial é suave e possui uma grande região convexa que vai cedendo lugar à platôs e mínimos locais mais íngremes à medida que a norma aumenta. A superfície de erro se tornará ainda mais complexa à medida que se aumentar a quantidade de dimensões do problema, o que dificultará o processo de otimização quando se utilizar algoritmos determinísticos baseados em direção de busca. Devido a esta dificuldade propõe-se uma estratégia para que seja possível manter as soluções iniciais do processo na região de interesse. Gera-se aleatoriamente valores de ângulos como solução inicial para o processo de otimização, em seguida inicia-se o processo de busca com um valor pequeno de norma. No passo seguinte, utiliza-se um valor maior de norma, mas mantem-se os valores de ângulos encontrados como solução final do processo anterior. Desta maneira, a solução encontrada numa superfície mais suave é projetada, no passo seguinte, para uma região próxima ao mínimo na nova superfície um

pouco menos suave, tornando fundamental a utilização de pequenos passos para incremento da norma durante o processo. Como pode-se ver na Figura 4.5 a superfície geral de erro mantem um padrão, sendo que mesmo se tornando mais complexa possuí um mesmo formato, isso faz com que com as projeções propostas mantenham as soluções encontradas para valores pequenos de norma na mesma região de interesse quando se utilizar valores maiores de norma. A figura 4.6 ilustra este processo para diferentes valores de norma, onde o erro mínimo foi encontrado para o primeiro valor de norma, e as demais curvas mostram a projeção desta solução.

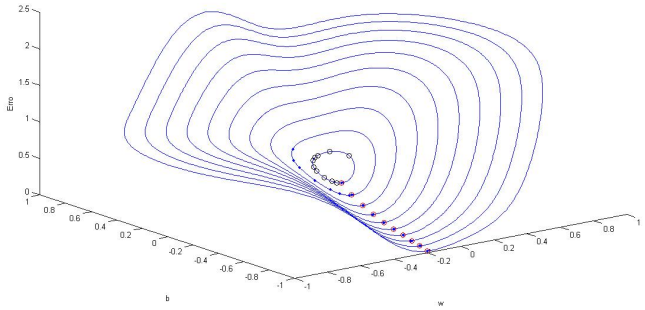

Figura 4.6: Valor de erro mínimo projetado para diversos valores de norma.

A utilização da formulação baseada em coordenadas hiperesféricas torna o problema de otimização da rede irrestrito, dado que o controle de complexidade está acoplado à função objetivo necessitando-se apenas utilizar o processo para diferentes valores de norma. Desta forma, pode-se utilizar qualquer algoritmo para otimização não linear para resolução do problema. Com o objetivo de gerar uma estimativa do conjunto pareto, será utilizado neste trabalho o método ϵ-restrito para o treinamento multi-objetivo da rede, da mesma forma que foi utilizado em [29].

4.3 Validação da Nova Formulação

Para validar o funcionamento da formulação da rede em função dos ângulos a partir da mudança no sistema de coordenadas, foram realizados testes iniciais utilizando-se dois algoritmos diferentes para o treinamento da rede. O primeiro algoritmo trata-se do tradicional *Back-propagation*, em sua versão original [32] e que possuí uma busca determinística através do gradiente descendente. O segundo algoritmo é o Evolução Diferencial (DE), um algoritmo evolucionário muito utilizado no âmbito da otimização não-linear com variáveis contínuas, o que faz com que ele seja adequado a este tipo de problema.

Desta maneira a formulação foi avaliada segundo algoritmos com filosofias diferentes para realização da busca pelo novo espaço de parâmetros.

Treinando uma rede neural de múltiplas camadas utilizando o algoritmo *Back-propagation* pode-se conseguir ajustar uma rede com esta formulação apenas acrescentando a derivada dos pesos em relação aos ângulos, devido ao emprego da regra da cadeia neste caso. Pode-se verificar que as derivadas acrescentadas são facilmente calculadas dado que a formulação do problema consiste basicamente na utilização das funções trigonométricas seno e cosseno, que têm derivadas simples e conhecidas. Desta forma a direção de ajuste dos ângulos é definida através de:

$$d = \frac{\partial e}{\partial \vec{\sigma}} \tag{4.4}$$

onde e é o erro médio quadrático da rede e $\vec{\sigma}$ representa o vetor de ângulos. Com isso tem-se a seguinte equação para se encontrar a i-ésima coordenada do vetor direção d:

$$\frac{\partial e}{\partial \sigma_i} = \frac{\partial e}{\partial \vec{w}} \frac{\partial \vec{w}}{\partial \sigma_i} \tag{4.5}$$

onde $\vec{w}$ é o vetor de pesos da rede. A busca para o novo valor de um determinado ângulo de forma que se minimize o erro quadrático da rede se dá da seguinte forma:

$$\sigma_i^{k+1} = \sigma_i^k - \alpha d \tag{4.6}$$

onde k é a iteração corrente do treinamento e α é o tamanho do passo na direção de busca.

A aplicação do DE como método de treinamento da rede consiste da utilização da versão original [36], onde foi possível aplicá-lo sem modificações em sua estrutura básica e, adicionando-se a função de avaliação do erro médio quadrático da rede como função objetivo. A operação mutação diferencial utilizada foi a descrita pela Equação 3.1. A validação realizada consiste na resolução de um problema de regressão onde utiliza-se a função seno com ruído gaussiano de média 0 e desvio padrão 0.2 para o treinamento da rede. Utilizou-se 100 amostras, sendo 70% para treinamento e 30% para teste. Os valores para norma variam de 0.5 a 10 com passo de 0.2. Utilizou-se 10 neurônios na camada escondida da rede e a solução escolhida foi definida por ser aquela com menor erro de validação, sendo que 20% dos dados de treinamento foram utilizados com este objetivo.

Com o treinamento utilizando o algoritmo baseado em gradiente não se

obteve uma boa estimativa do conjunto pareto.

A Figura 4.7 mostra o comportamento do erro quadrático médio (EQM) durante as iterações do algoritmo usando o gradiente. Nota-se que possivelmente existem inconsistências no treinamento dado que com o aumento da norma o erro perde a tendência de decrescimento. É provavel que este comportamento indesejado é devido à grande interconexão entre os parâmetros da rede nesta formulação, o que pode dificultar a utilização do *back-propagation* clássico dado que a modificação de um determinado ângulo pode influenciar em até todos os pesos da rede, tornando o processo de otimização mais sensível do que se fosse utilizado o espaço definido pelos pesos de forma direta.

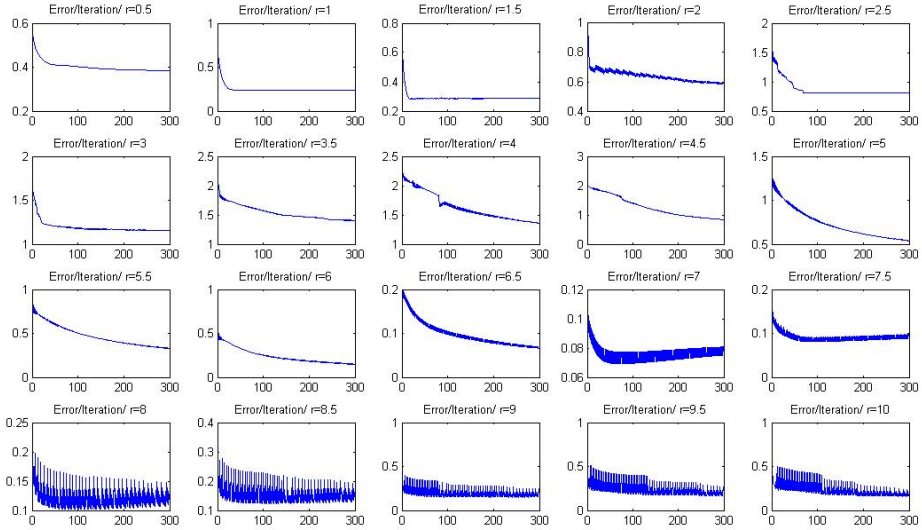

Figura 4.7: Comportamento do erro durante as iterações do *Back-Propagation* para cada valor de norma.

O histograma dos pesos da rede, levando em consideração a solução selecionada, pode ser visualizado pela Figura 4.8, onde nota-se que alguns valores ficam um pouco mais afastados da tendência central, provavelmente, estes são os pesos tomados como referência durante a modelagem do problema.

A aproximação para a função seno, obtida pela solução escolhida no processo decisório, pode ser visualizada através da Figura 4.9, onde a linha vermelha representa o modelo obtido pela rede.

O treinamento do NSS utilizando o algoritmo DE mostrou resultados muito promissores com uma estimativa aceitável do conjunto pareto gerado e um melhor resultado quanto a aproximação da função. Utilizou-se para o DE uma população com 60 soluções candidatas e 100 iterações. Os parâmetros utilizados para a operação de mutação diferencial foram $\eta = 0,99$, $\lambda = 0.99$, o que faz com que o algoritmo praticamente não tenha influência dos parâmetros,

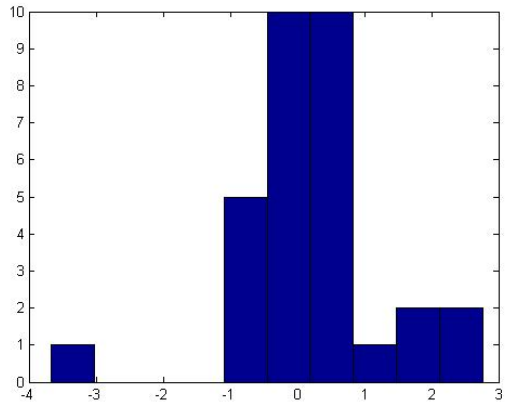

Figura 4.8: Histograma dos pesos na solução selecionada utilizando o *Back-Propagation*.

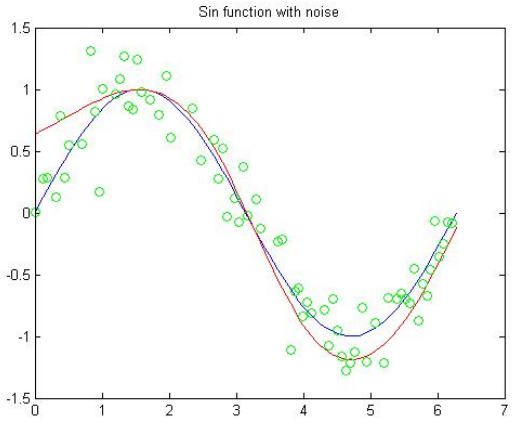

Figura 4.9: Aproximação da função seno obtida pelo modelo selecionado, utilizando o *Back-Propagation*.

utilizando apenas a sua característica de auto adaptação. As Figuras 4.10, 4.11, 4.12 e 4.13 mostram a estimativa do pareto, comportamento do erro, histograma dos pesos e aproximação para a função seno, respectivamente.

O DE mostrou resultados estáveis e boa adaptação à superfície gerada a partir da formulação no sistema de coordenadas proposto.

4.4 Considerações finais

Neste capítulo detalhou-se a formulação baseada em coordenadas hiperesféricas para representação do problema de otimização em redes neurais. Foi mostrada uma proposta de treinamento da rede utilizando a nova formulação, sendo que esta foi validada com a utilização de dois algoritmos de otimização. Como resultado dos testes verificou-se que a nova formulação é consistente

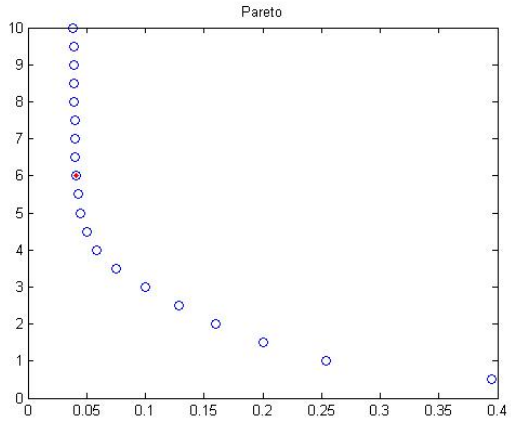

Figura 4.10: Estimativa do pareto obtido após o treinamento do NSS utilizando o DE.

e promissora, mostrando bons resultados no problema de regressão avaliado. Vistos os resultados da aplicação inicial do método o algoritmo DE será utilizado ao longo do trabalho como algoritmo de treinamento da rede, devido aos bons resultados obtidos durante a avaliação da formulação.

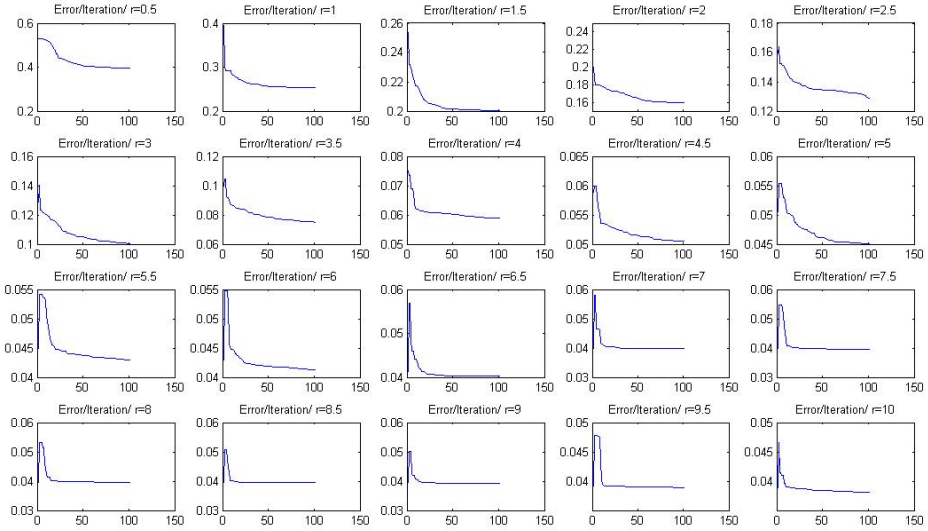

Figura 4.11: Comportamento do erro durante as iterações do DE para cada valor de norma.

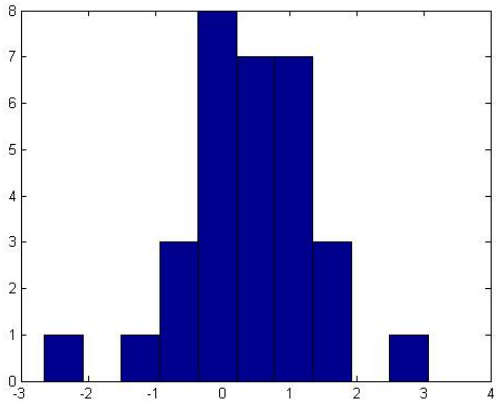

Figura 4.12: Histograma dos pesos na solução selecionada utilizando o DE.

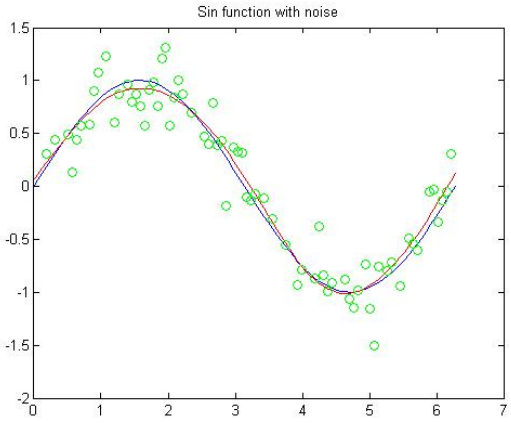

Figura 4.13: Aproximação da função seno obtida pelo modelo selecionado utilizando o DE.

CAPÍTULO 5

Metodologia para Simulações e Testes

Nos capítulos anteriores foram apresentados os métodos e abordagens que definem as contribuições deste trabalho. Este capítulo detalha a metodologia utilizada para testar os métodos propostos e apresenta as bases de dados e problemas a serem resolvidos. Os testes consistem na aplicação dos métodos para treinamento multi-objetivo de RNAs a problemas de regressão e classificação. A abordagem proposta para seleção multi-objetivo de características é aplicada a uma base de dados com grande dimensionalidade e poucas amostras. Os resultados destes testes serão apresentados no capítulo seguinte.

5.1 Metodologia de testes para o DEANN e o NSS

Para avaliação da capacidade de generalização do DEANN e do NSS foram utilizados quatro problemas de regressão e três problemas de classificação de padrões. O método MOBJ [29] [38] foi tomado como base para comparações, visto que este é um algoritmo eficiente para busca de modelos neurais com boa capacidade de generalização, além de ser baseado no controle da complexidade aparente da rede, analogamente aos métodos propostos. O decisor utilizado para seleção do modelo dentro do conjunto pareto estimado pelos métodos é o baseado no menor erro de validação, que consiste na avaliação dos modelos obtidos aplicados a um conjunto de dados, não conhecidos durante o treinamento, definido como conjunto de validação. Após a escolha do

decisor, os dados do conjunto definido como teste foram utilizados para avaliar a qualidade do modelo. Os critérios utilizados para avaliação foram o erro médio quadrático (EMQ) e acurácia para os problemas de regressão e classificação respectivamente e o tempo médio de execução (TME) em ambos os tipos de problema, definido em segundos. Foram realizadas 10 execuções de cada algoritmo para validação estatística dos métodos, onde tomou-se ao fim a média e desvio padrão para análise baseada nos critérios de avaliação definidos. As simulações foram realizadas num computador com processador Intel Core 2 duo 2.1GHz com 4GB de memória, sistema operacional Windows Seven e o software Matlab R2010a. Em todos os métodos utilizados, levando-se em consideração os problemas de regressão, as camadas escondida e de saída da rede utilizaram função de ativação do tipo tangente hiperbólica e linear, respectivamente, sendo que para os problemas de classificação a função sigmoidal foi utilizada como saída da rede. A arquitetura da MLP para todos os problemas foi definida como 1-10-1, visando o treinamento com uma rede de estrutura sobre-parametrizada, visando a obtenção de um conjunto de soluções que varie da menos complexa até a mais complexa como resultado dos métodos após o treinamento. Nas simulações realizadas foram utilizados problemas de regressão tipicamente encontrados na literatura para este propósito. As funções que descrevem cada problema são:

$$
\begin{aligned}
f1(x) &= sine(x) \\
f2(x) &= 4,26(e^{-x} - 4e^{-2x} + 3e^{-3x}) \\
f3(x) &= (x-2)(2x+1)/(1+x^2) \\
f4(x) &= (e^{-0,2x}) + (2e^{-0,2x} * sine(2\pi * 0,2x - \pi/4) - 0,27)
\end{aligned}
$$

Os conjuntos de dados gerados para aplicação dos métodos consistem de 180 amostras, sendo 100 atribuídas ao conjunto de treinamento, 50 ao conjunto de validação e 30 pertencentes ao conjunto de testes. As observações geradas são limitadas nos intervalos $[0; 2\pi], [0; 3, 25], [-8; 12] e [0, 10]$ para as funções $f1$, $f2$, $f3$ e $f4$ respectivamente e obtidas a partir das funções geradoras com o acréscimo de um ruído gaussiano com média 0 e desvio padrão 0,2. A variação de ϵ nos métodos foi de 0,5 a 10, sendo que estes limites foram obtidos através de observações nos resultados para obtenção das soluções extremas do pareto. Variando-se nestes limites como o passo de 0,2 obteve-se um total de 48 soluções. As bases de dados utilizadas para classificação são todas binárias e foram obtidas em [4]. A primeira é a base diabetes da Índia, que contem 768 amostras e 8 atributos. A segunda base é a do câncer de mama de Wisconsin, que consiste de 569 amostras e 32 atributos. A ter-

ceira base é a de doenças do coração, que é formada por 270 padrões e 13 atributos. As bases de dados foram divididas de forma que utilizou-se 50% dos dados para treinamento, 25% para validação e 25% para testes, sendo que estas amostragens foram realizadas de maneira aleatória e mantendo-se a proporção entre as classes. O valor de ϵ para os problemas de classificação foi definido na faixa de 0,1 a 6, o que resultou num conjunto com 30 soluções utilizando-se um passo de 0,2. Definidos os parâmetros referentes às bases de dados para os problemas de regressão e classificação o próximo passo foi a definição dos parâmetros relativos a cada método avaliado. Os dois métodos propostos utilizam o algoritmo DE para o treinamento da rede, diferenciando-se na superfície de busca gerada pela formulação do problema. Desta maneira os parâmetros de ambos foram os mesmos, consistindo dos seguintes valores: $N_p = 60$, $N_{gen} = 30$, $\lambda = 0,99$, $\eta = 0,99$ e $C = 0,9$, onde N_p é a quantidade indivíduos na população e $N_{gen} = 30$ é quantidade de iterações do algoritmo. Os parâmetros $\lambda = 0,99$, $\eta = 0,99$ e $C = 0,9$ foram definidos empiricamente após alguns testes.

5.2 Aplicação da seleção clonal de características

Na base de dados a seguir é aconselhável a utilização da abordagem para seleção de características proposta no capítulo 3 para tornar mais fácil a tarefa de classificação. Esta base é aqui utilizada para testar e avaliar a abordagem proposta.

5.2.1 Base de dados

A base de dados utilizada é procedente do trabalho apresentado em [16] e consiste de expressões gênicas de 7129 sondas referentes a 6817 genes humanos e 72 amostras de dados referentes à pacientes com leucemia aguda linfóide (em inglês, *acute lymphoblastic leukemia* - ALL) e leucemia aguda mielóide (em inglês, *acute myeloid leukemia* - AML). Estas amostras foram divididas em dois conjuntos, onde 38 destas (27 ALL, 11 AML), provenientes de medula óssea, foram definidas como conjunto de treinamento. As outras 34 amostras (20 ALL, 14 AML) foram definidas como conjunto independente, sendo que 24 destas foram obtidas da medula óssea e as outras 10 de sangue periférico.

5.2.2 Seleção clonal

Inicialmente aplicou-se os filtros *F-Score* e *Pearson* para rankear as 7129 características presentes na base de dados da leucemia. Após a obtenção deste rank as 50 primeiras características obtidas em cada método foram pré-selecionadas para serem submetidas ao algoritmo Clonal. Para o treinamento do classificador de Bayes utilizou-se o conjunto de treinamento com 38 amostras, enquanto que o conjunto independente foi dividido proporcionalmente pela metade em dois subconjuntos, validação e teste. Após a obtenção do subconjunto de características mais adequado à classificação testou-se este subconjunto com o próprio classificador de Bayes e com o K-NN. Utilizou-se também o algoritmo *K-means* [12] para geração de clusters obtidos a partir dos dados de teste usando o subconjunto de características selecionado, com o objetivo de visualizar de forma geométrica a disposição dos padrões. Para o algoritmo Clonal utilizou-se uma população com 80 anticorpos e 100 gerações foram definidas como critério de parada. Os demais parâmetros utilizados foram: b = 80%, d = 20%, ρ = 3,2 e β = 0,5. O valor 0,7 foi utilizado para o parâmetro w na ponderação dos objetivos na função afinidade, dando mais prioridade à taxa de classificações corretas do que para o tamanho do subconjunto de características.

CAPÍTULO 6

Resultados

Apresenta-se neste capítulo os resultados obtidos na aplicação dos métodos multi-objetivo para o treinamento de rede aos problemas de regressão e classificação. São exibidos também os resultados obtidos com a aplicação da abordagem para seleção multi-objetivo de características à base de dados da leucemia.

6.1 Simulações para o treinamento multi-objetivo de rede

As simulações apresentadas nesta seção mostram a capacidade de aprendizagem e generalização dos métodos para treinamento multi-objetivo de RNAs apresentados nos capítulos 3 e 4 na aplicação em problemas teste.

6.1.1 Problemas de regressão

Foram realizadas simulações para as quatro funções de regressão e a seguir são exibidos resultados que demonstram o grau de aproximação obtido para cada um dos métodos avaliados além de estimativas do conjunto pareto-ótimo obtidas em cada simulação.

Nas Figuras 6.1, 6.2, 6.3 e 6.4 podem ser visualizadas as estimativas de conjuntos pareto geradas para as funções $f1$, $f2$, $f3$ e $f4$ respectivamente, com a utilização de cada método.

Nas Figuras 6.5, 6.6, 6.7 e 6.8 são mostrados os modelos obtidos para as funções $f1$, $f2$, $f3$ e $f4$ para cada método testado, sendo também apresentados

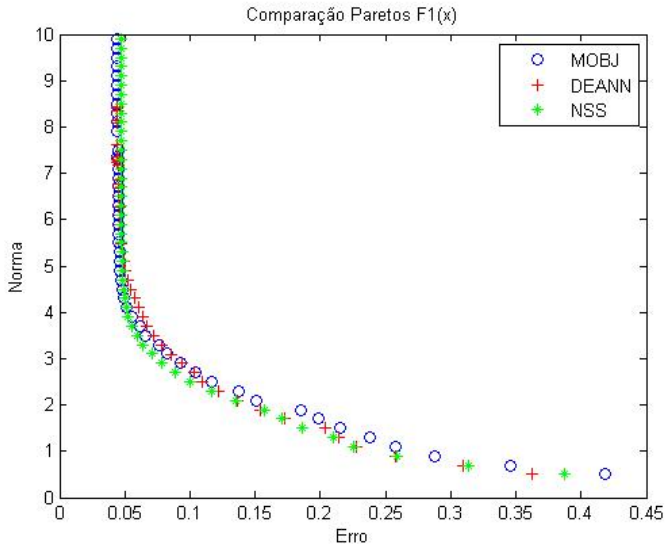

Figura 6.1: Estimativa do pareto para a função $f1$ após o treinamento com cada método.

na figura a função geradora e os dados de treinamento utilizados no processo. As curvas exibidas são resultantes de soluções escolhidas pelo decisor de validação.

A tabela 6.1 mostra o erro médio quadrático para os dados de teste obtidos para cada base de dados de regressão, considerando-se todos os métodos avaliados. Pode-se visualizar nesta tabela a média(Md) do erro, o desvio padrão(Dp) e os valores mínimo(Mn) e máximo(Mx), dadas as execuções realizadas. Na tabela 6.2 pode ser visualizado o tempo médio de execução dos métodos avaliados para todas as bases de regressão, considerando-se as mesmas medidas estatísticas utilizadas na avaliação do erro.

6.1.2 Problemas de classificação

Nas simulações realizadas para a tarefa de classificação são gerados os mesmos indicadores vistos para os problemas de regressão, apenas substituindo-se os indicadores de erro pela acurácia para os dados de teste, sendo esta uma medida amplamente utilizada para avaliação do desempenho de classificadores. São mostradas também nesta seção, as estimativas do conjunto pareto-ótimo obtidas após o treinamento de todos os métodos para cada base de dados.

Observa-se que as dimensões do problema de otimização da rede variam de acordo com a quantidade de atributos inerente a cada base de dados, devido à estrutura da rede mudar de acordo com as conexões de entrada. Desta maneira, considerando-se os 10 neurônios definidos para a camada escondida da

Tabela 6.1: EQM obtido para os problemas de regressão utilizando cada método

		MOBJ	**DEANN**	**NSS**
$f1$	**Md**	$0,0383$	$0,0363$	$0,0361$
	Dp	$0,0005$	$0,0009$	$0,0024$
	Mn	$0,0375$	$0,0349$	$0,0332$
	Mx	$0,0389$	$0,0381$	$0,0395$
$f2$	**Md**	$0,0330$	$0,0402$	$0,0373$
	Dp	$0,0030$	$0,0015$	$0,0073$
	Mn	$0,0295$	$0,0382$	$0,0314$
	Mx	$0,0395$	$0,0424$	$0,0549$
$f3$	**Md**	$0,0369$	$0,0375$	$0,0368$
	Dp	$0,0003$	$0,0005$	$0,0012$
	Mn	$0,0363$	$0,0367$	$0,0342$
	Mx	$0,0378$	$0,0384$	$0,0377$
$f4$	**Md**	$0,0387$	$0,0448$	$0,0480$
	Dp	$0,0022$	$0,0025$	$0,0021$
	Mn	$0,0349$	$0,0406$	$0,0450$
	Mx	$0,0429$	$0,0494$	$0,0510$

Tabela 6.2: TME obtido para os problemas de regressão utilizando cada método

		MOBJ	**DEANN**	**NSS**
$f1$	**Md**	$8,1949$	$9,1510$	$162,8274$
	Dp	$0,6488$	$28,9381$	$10,9319$
	Mn	$7,1384$	$2,7628$	$155,6338$
	Mx	$9,3606$	$91,5102$	$187,5085$
$f2$	**Md**	$8,1956$	$9,5036$	$160,5251$
	Dp	$0,4759$	$30,0530$	$0,4773$
	Mn	$7,5986$	$1,2313$	$159,8014$
	Mx	$8,9573$	$95,0398$	$161,4952$
$f3$	**Md**	$8,0325$	$9,6034$	$161,0197$
	Dp	$0,3023$	$30,3687$	$0,4333$
	Mn	$7,6093$	$1,1287$	$160,1391$
	Mx	$8,5076$	$97,1342$	$161,7486$
$f4$	**Md**	$7,2227$	$8,8031$	$157,3960$
	Dp	$0,7501$	$27,8380$	$2,3180$
	Mn	$6,5334$	$1,0126$	$154,6125$
	Mx	$9,0250$	$89,4314$	$160,8469$

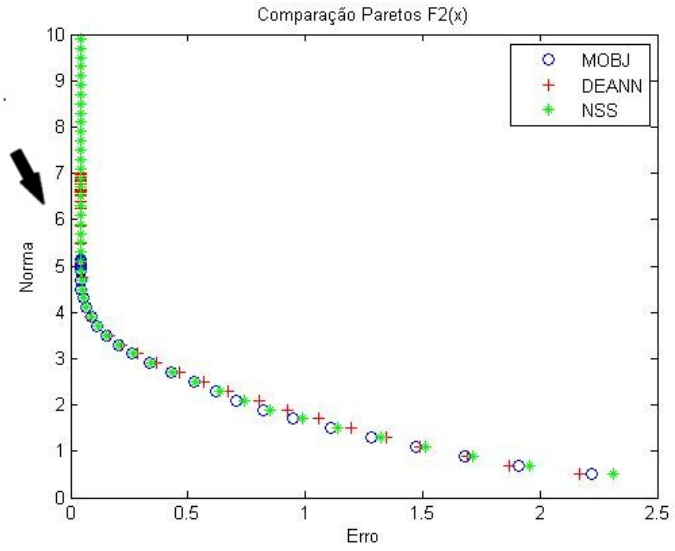

Figura 6.2: Estimativa do pareto para a função $f2$ após o treinamento com cada método.

rede e 1 neurônio para a camada de saída, existem 101 variáveis de otimização para a base de dados da diabetes, 331 para a base do câncer de mama e 151 para a base de doenças do coração.

As tabelas 6.3 e 6.4 exibem os indicadores relativos à acurácia e tempo médio de execução para todas as bases de dados utilizando cada método.

As figuras 6.9, 6.10 e 6.11 mostram estimativas para o conjunto pareto obtidas através da aplicação dos métodos a cada base de dados.

6.1.3 Análise de resultados

Os problemas de regressão utilizados nas simulações possuem apenas uma variável de entrada, o que torna o treinamento menos custoso computacionalmente aos métodos se comparado às bases de classificação. Através da Tabela 6.1 é verificado que a média do erro foi bem semelhante em todos os métodos mostrando uma equivalência dos métodos avaliados no quesito eficácia. Em relação ao tempo computacional, verifica-se através da Tabela 6.2 que o método DEANN foi equivalente ao MOBJ, ambos obtiveram tempos bem próximos, no entanto o DEANN teve um desvio padrão elevado, o que pode ter ocorrido devido à natureza estocástica do treinamento. O NSS neste caso foi o pior método devido ao alto custo computacional encontrado. Diversos fatores podem ter influenciado para o alto custo computacional do NSS, desde os parâmetros utilizados no algoritmo evolutivo de treinamento até o aumento de complexidade no cálculo de função devido à camada de complexidade adicional inerente à formulação introduzida por ele.

Tabela 6.3: Acurácia obtida para as bases de classificação utilizando cada método

		MOBJ	DEANN	NSS
Diabetes	**Md**	0.7491	0.7783	0.7765
	Dp	0.0036	0.0065	0.0062
	Mn	0.7435	0.7696	0.7609
	Mx	0.7522	0.7913	0.7826
Câncer	**Md**	0.9882	0.9512	0.9819
	Dp	0.0000	0.0028	0.0033
	Mn	0.9882	0.9471	0.9766
	Mx	0.9882	0.9529	0.9883
Coração	**Md**	0.8259	0.8642	0.8889
	Dp	0.0039	0.0058	0.0101
	Mn	0.8148	0.8519	0.8765
	Mx	0.8272	0.8765	0.9012

Tabela 6.4: TME obtido para as bases de classificação utilizando cada método

		MOBJ	DEANN	NSS
Diabetes	**Md**	2.0582	18.8480	64.5260
	Dp	0.1607	59.6027	12.2158
	Mn	1.8571	1.1254	46.2044
	Mx	2.4168	189.5804	84.2729
Câncer	**Md**	9.8125	23.6872	168.1766
	Dp	1.0284	74.9055	4.4819
	Mn	8.1842	2.3471	165.4678
	Mx	11.3090	239.4719	178.5135
Coração	**Md**	1.5940	15.3864	75.4833
	Dp	0.2328	48.6560	0.3841
	Mn	1.3763	2.3710	75.0370
	Mx	2.0017	156.1638	76.0353

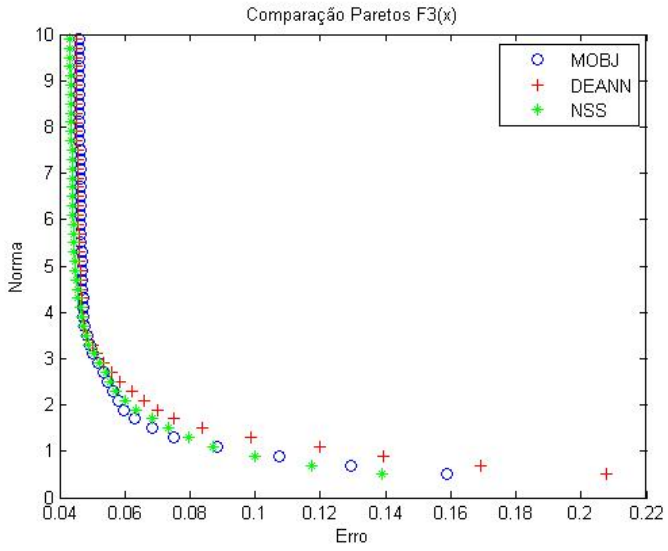

Figura 6.3: Estimativa do pareto para a função $f3$ após o treinamento com cada método.

Através das figuras 6.5 - 6.8 verifica-se que os métodos também foram equivalentes nos modelos escolhidos. Todos se aproximam da função geradora, analisando-se apenas de forma visual.

Em relação aos conjuntos pareto obtidos para os problemas de regressão, pode-se verificar por meio das figuras 6.1 - 6.4 que todos os métodos geraram estimativas semelhantes. É interessante observar que na Figura 6.2 o pareto gerado pelo MOBJ não conseguiu progredir a partir de um certo ponto, ficando preso próximo ao centro do gráfico considerando-se as soluções extremas definidas. O DEANN conseguiu ir um pouco mais a frente gerando soluções de norma mais elevada enquanto que o NSS conseguiu um resolução melhor que os outros, gerando uma estimativa que conseguiu varrer todo o espaço de soluções. Este fato pode ter ocorrido devido a uma limitação imposta pela função $f2$, mas que não impediu a geração de um pareto com boa resolução pelo NSS devido à formulação utilizada que garante uma busca pelo erro sem variação da norma definida.

Nas simulações para as tarefas de classificação, pode-se ver através da tabela 6.3 que os métodos propostos obtiveram maior valor de acurácia do que o MOBJ, exceto para a base do câncer. Este resultado pode ter ocorrido devido à utilização do algoritmo DE no treinamento do DEANN e do NSS, que torna o problema mais difícil de ser resolvido devido à base de dados do câncer ter muitas dimensões, gerando elementos maiores e mais difíceis de se trabalhar nos algoritmos evolutivos. Em relação ao maior valor de acurácia obtido, o MOBJ não superou nenhum dos métodos propostos. Quanto ao tempo computacional o MOBJ superou o DEANN e o NSS, sendo que a utilização do DE

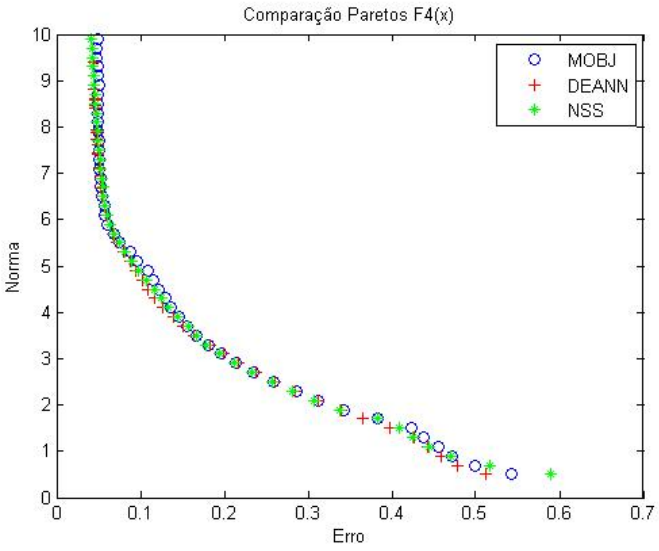

Figura 6.4: Estimativa do pareto para a função $f4$ após o treinamento com cada método.

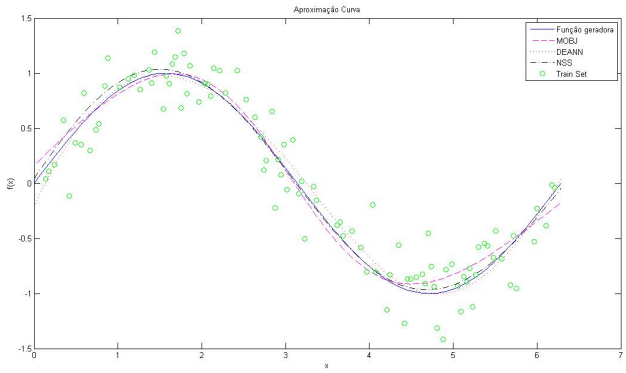

Figura 6.5: Aproximação para a função $f1$ após o treinamento com cada método.

para o treinamento dos métodos pode ter contribuído para este resultado. Nas figuras 6.9, 6.10 e 6.11 podem ser visualizados os conjuntos pareto gerados por cada método, o que mostra semelhança entre as estimativas obtidas.

6.2 Resultados da aplicação da abordagem para seleção de características

Após o rankeamento das sondas através dos dois filtros uni-variados utilizados, as primeiras 50 sondas encontradas por cada método foram submetidas ao algoritmo Clonal, que retornou um subconjunto de 16 sondas referentes ao *F-Score* (S1) e um subconjunto de 15 sondas referentes ao método de *Pearson* (S2). Após a seleção realizada pelo clonal gerou-se um novo sub-

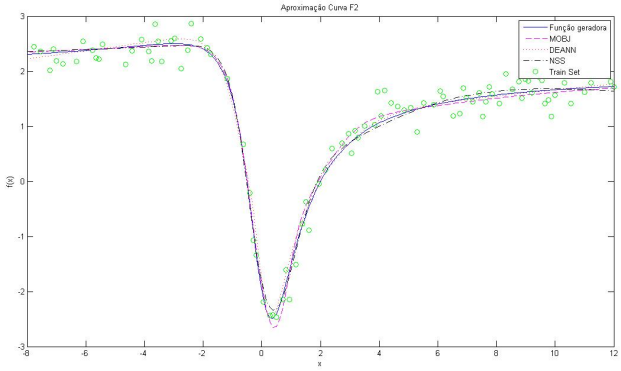

Figura 6.6: Aproximação para a função $f2$ após o treinamento com cada método.

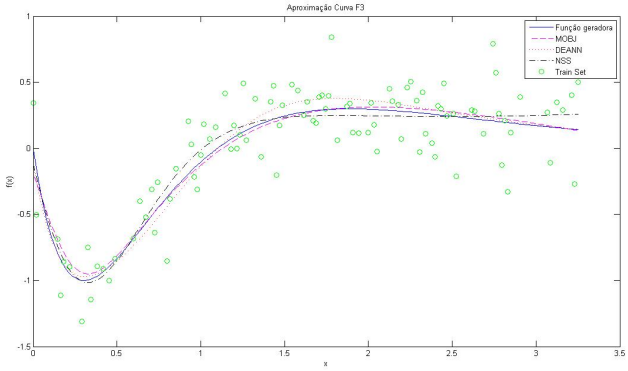

Figura 6.7: Aproximação para a função $f3$ após o treinamento com cada método.

conjunto S3 contendo as sondas que aparecem tanto em S1 quanto em S2. A Tabela 6.5 mostra a relação de sondas dos conjuntos S1, S2 e S3, indicando o índice de cada sonda dentre as 7129.

Tabela 6.5: Relação de sondas em cada subconjunto

	Sondas pré-selecionadas
S1	2020, 2288, 3847, 1882, 4196, 2402, 6200, 1674, 6803, 1807, 3605, 6405, 5808, 2001, 4377, 6919
S2	3320, 2020, 5039, 1834, 4196, 2288, 6201, 1882, 2121, 6803, 2402, 3605, 6677, 6405, 4377
S3	1882, 2020, 2288, 2402, 3605, 4196, 4377, 6405, 6803

As Tabelas 6.6 e 6.7 mostram a taxa classificações corretas para os dados de teste e conjunto independente (validação + teste) respectivamente, utilizando as sondas definidas em S1, S2 e S3 para o classificador de *Bayes* e o K-NN.

Através das Figuras 6.12 e 6.13 podem ser visualizados os clusters formados pelo algoritmo K-means com o conjunto de dados de teste utilizando os subconjuntos S1 e S2 respectivamente, onde considerando estes clusters,

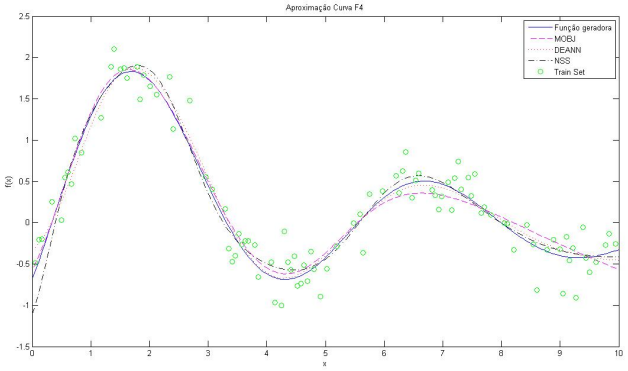

Figura 6.8: Aproximação para a função $f4$ após o treinamento com cada método.

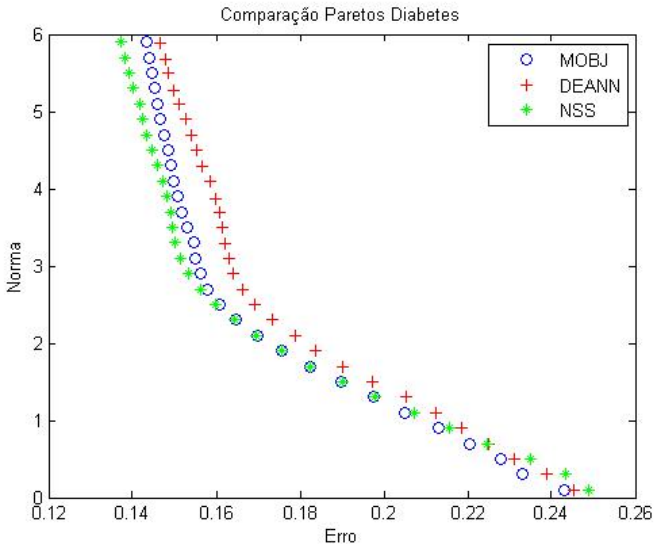

Figura 6.9: Estimativa do pareto para a base da diabetes após o treinamento com cada método.

Tabela 6.6: Percentual de classificações corretas para o conjunto de dados de teste

	Bayes	**KNN**
S1	$94,1176\%$	$88,2353\%$
S2	$94,1176\%$	$94,1176\%$
S3	100%	$94,1176\%$

obteve-se 94,1176% dos padrões agrupados em suas classes corretas para ambos os subconjuntos.

Nas Figuras 6.14 e 6.15 são mostrados os clusters gerados pelo K-means para o subconjunto S3 utilizando o conjunto de dados de teste e conjunto total de dados (treinamento + independente) respectivamente. Neste caso obteve-se

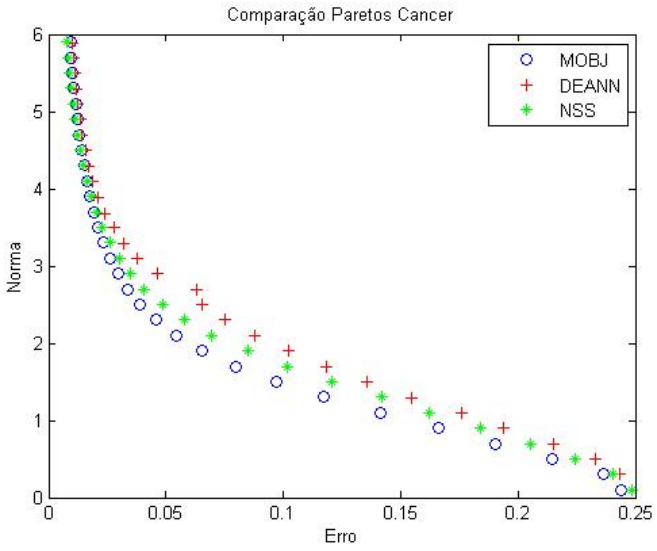

Figura 6.10: Estimativa do pareto para a base do câncer após o treinamento com cada método.

Tabela 6.7: Percentual de classificações corretas para o conjunto de dados de independente

	Bayes	**KNN**
S1	$97,0588\%$	$82,3529\%$
S2	$97,0588\%$	$85,2941\%$
S3	100%	$91,1765\%$

94,1176% dos dados de teste agrupados em suas classes corretas e 91,6667% para o conjunto de dados total. Os clusters gerados pelo K-means são visualizados em relação às duas primeiras sondas de cada subconjunto.

Através dos resultados apresentados pode-se visualizar que os subconjuntos S1 e S2 têm efeitos muito semelhantes, sendo iguais quando se utilizou o classificador de *Bayes* tanto para o conjunto de testes como para o conjunto de dados independente e, verificou-se uma leve superioridade de S2 nos mesmos casos utilizando o K-NN. Quando se utilizou o conjunto S3 verificou-se resultados superiores em todos os casos em relação aos outros subconjuntos.

6.3 Considerações finais

Neste capítulo os métodos multi-objetivo propostos para o treinamento de RNAs foram aplicados a problemas de regressão e classificação com diferentes características e dimensões, obtendo bons resultados quanto ao EQM e acurácia mas com custo computacional relativamente elevado principalmente

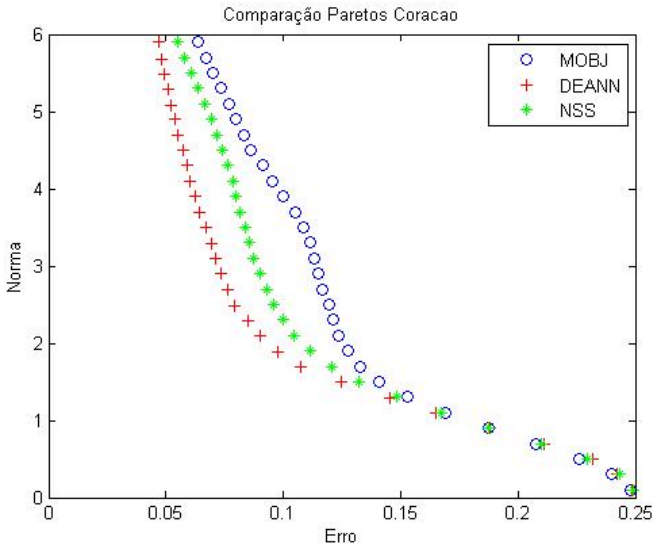

Figura 6.11: Estimativa do pareto para a base de doenças do coração após o treinamento com cada método.

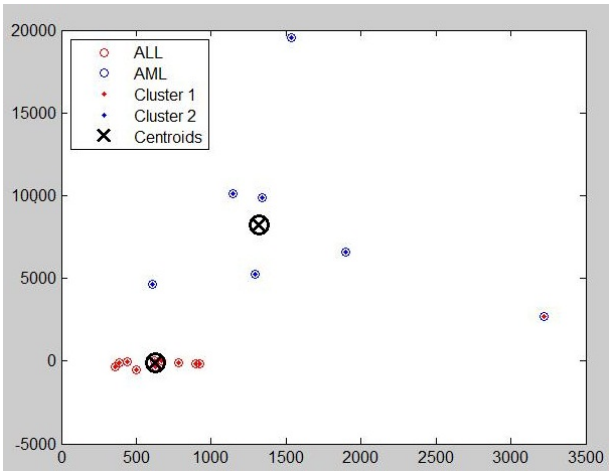

Figura 6.12: Clusters gerados pelo K-means para o conjunto teste utilizando S1

nos problemas de classificação. Os fatores determinantes para os resultados foram discutidos e o algoritmo DE utilizado no treinamento dos métodos propostos foi considerado o principal responsável pelo custo computacional elevado. Foi avaliado também a aplicação da abordagem para seleção de características à base de dados da leucemia cuja dimensionalidade é elevada. Foi possível verificar uma redução considerável do conjunto de atributos disponíveis e foram obtidos bons resultados quanto a acurácia para os classificadores utilizados para teste.

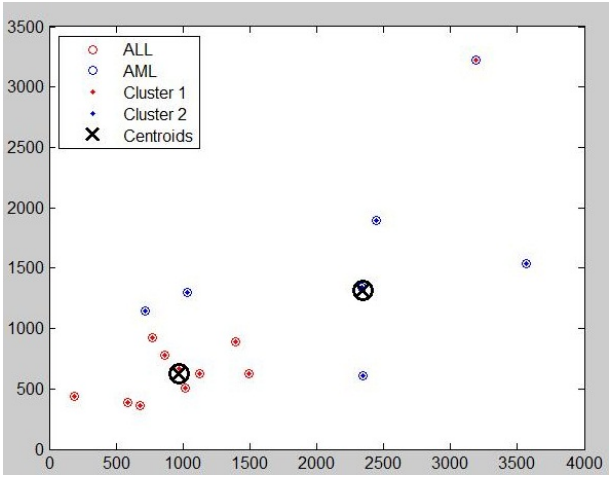

Figura 6.13: Clusters gerados pelo K-means para o conjunto teste utilizando S2

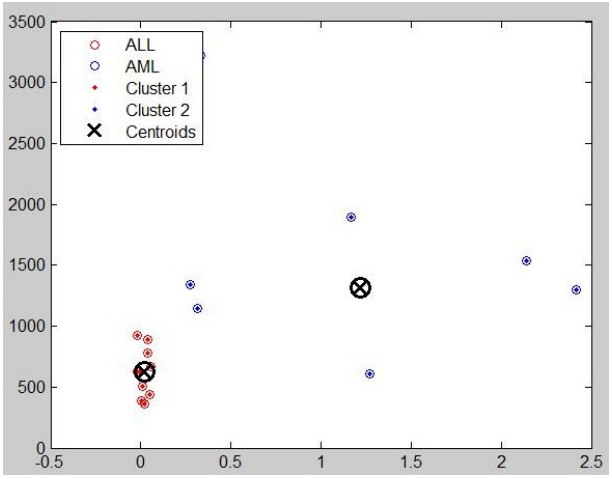

Figura 6.14: Clusters gerados pelo K-means para o conjunto teste utilizando S3

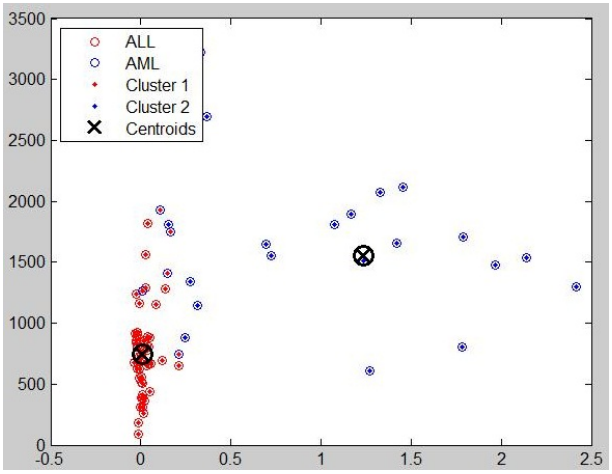

Figura 6.15: Clusters gerados pelo K-means para o conjunto total utilizando S3

CAPÍTULO 7

Conclusões

Neste trabalho foram desenvolvidos dois novos métodos para o treinamento multi-objetivo de RNAs e uma nova abordagem para seleção multi-objetivo de características. Os métodos para treinamento de RNAs foram comparados ao método MOBJ e visam uma forma de treinamento diferente, de maneira que se evite o tratamento de restrições de forma direta no algoritmo de otimização utilizado para o treinamento da rede. O método DEANN utiliza uma forma de acoplar esta restrição aos limites dos pesos da rede, o que facilita muito a implementação do método que tem como princípio o treinamento da rede utilizando o DE. O DEANN mostrou eficácia como pode ser visto na seção 6, tendo obtido resultados muito promissores, necessitando apenas que se pesquise formas para diminuição do custo computacional. Em relação ao método NSS, considerou-se muito promissores os resultados encontrados devido à eficácia obtida nas simulações realizadas, sendo que custo computacional também foi a principal dificuldade neste método.

O NSS além de ter obtido bons resultados nas simulações trouxe a adição de novas características, promissoras para o treinamento de RNAs. A utilização do sistema de coordenadas hiperesféricas para representação do problema de otimização da rede gerou uma nova região de busca que deve ser melhor explorada. Os maus resultados no quesito custo computacional podem ser decorrentes do algoritmo evolutivo utilizado para o treinamento, o que traz a possibilidade de que utilizando-se outros métodos de otimização não linear irrestrita, o que é possível devido à natureza da nova formulação, possam ser obtidos melhores resultados neste sentido. Métodos determinísti-

cos podem ser interessantes para o treinamento do método pois mesmo não se encontrando bons resultados com os testes realizados com o *back-propagation* simples, a facilidade em se obter as derivadas facilita a utilização deste tipo de método para o treinamento, necessitando-se de uma busca extensiva para tentar se encontrar um método adequado à superfície de busca do problema gerado. Em relação à geração das estimativas do conjunto pareto-ótimo, o NSS mostrou-se mais consistente que os outros analisados, sendo que ao garantir que cada solução permanecerá em uma região limitada pela norma, faz com que a única preocupação no treinamento seja a minimização do erro, fazendo com que a resolução do conjunto pareto no que diz respeito a um dos objetivos seja facilmente controlada. Através dos resultados obtidos verifica-se que na maioria dos casos os métodos propostos obtiveram estimativas do conjunto pareto com boa resolução, o que conduz à um processo decisório mais consistente.

Em relação à abordagem para seleção multi-objetivo de características, foi apresentado um método híbrido de seleção de características que realiza uma pré-seleção com filtros uni-variados e uma seleção multivariada através de um método *wrapper*. Na seleção multivariada utiliza-se o algoritmo Clonal como estratégia de busca e o classificador de Bayes para avaliação dos subconjuntos de características. A classificação de tipos de leucemia em ALL e AML teve bons resultados com a utilização dos subconjuntos de sondas selecionadas pelo método utilizado. Percebeu-se que as melhores taxas de classificação foram alcançadas quando se combinou os subconjuntos resultantes retornados pelo método *wrapper*.

Como propostas para trabalhos futuros deixa-se algumas sugestões. Em relação ao DEANN sugere-se que pesquisas mais extensivas sejam realizadas na utilização da abordagem, de forma que variações do DE utilizado podem conduzir a melhores resultados quanto ao tempo computacional. Outras melhorias poderiam ocorrer com a mudança na forma de tratamento do problema multi-objetivo, utilizando-se outras formas de se transformar o problema ou com a utilização de um DE multi-objetivo baseado em pareto-dominância para obtenção das soluções.

Em relação ao NSS sugere-se a utilização de novos algoritmos para o treinamento. Métodos determinísticos deveriam ser melhor explorados devido à natureza da formulação facilitar a utilização destes. Possíveis dificuldades geradas pela camada de complexidade adicionada à superfície de busca poderiam ser minimizadas com a utilização de métodos que visam a convexificação desta superfície como visto em [3] [25], o que facilitaria a utilização de métodos determinísticos que tem como premissa que a região seja convexa para que o

processo de otimização seja eficiente.

Quanto à abordagem para seleção de características a utilização de outros filtros uni-variados ou mesmo multivariados para pré-seleção das características poderia trazer maiores parâmetros para comparações. A combinação dos dados obtidos pelos filtros uni-variados num momento anterior à submissão ao método *wrapper* poderia gerar melhores resultados. A inclusão de novos tipos de gráficos para visualização da dispersão dos dados e agrupamento entre classes e a utilização de outros classificadores para avaliação das características selecionadas enriqueceriam trabalhos posteriores.

Referências Bibliográficas

[1] H. A. ANTON. *Cálculo*, volume 2. Bookman, 2007.

[2] P. Bartlet. For valid generalization the size of the weights is more important than the size of the network. *Advances in Neural Information Processing Systems*, 9:134–140, 1997.

[3] D. P. Bertsekas. Convexification procedures and decomposition methods for nonconvex optimization problems. *Journal of Optimization Theory and Applications*, 29(2):169–197, 1979.

[4] C. L. Blake and C. J. Merz. Uci repository of machine learning databases, 1998.

[5] A. P. Braga, T. B. Ludemir, and A. C. P. F. Carvalho. *Redes Neurais Artificiais. Teoria e aplicações. 2.ed.* LTC Editora, Rio de Janeiro, 2007.

[6] Y. Chang and C. Lin. Feature ranking using linear svm. *WCCI2008 Workshop and Conference Proceedings*, 3:53–64, 2008.

[7] S. Cho. Exploring features and classifiers to classify gene expression profiles of acute leukemia. *International Journal of Pattern Recognition and Artificial Intelligence*, 16:831–844, 2002.

[8] M. A. Costa, A. P. Braga, B. R. de Menezes, R. Teixeira, and G. Parma. Training neural networks with a multi-objective sliding mode control algorithm. *Neurocomputing*, 51:467–473, 2002.

[9] L. N. De Castro. *Engenharia Imunológica: Desenvolvimento e Aplicação de Ferramentas Computacionais Inspiradas em Sistemas Imunológicos Artificiais*. Tese de doutorado, Faculdade de Engenharia Elétrica e de Computação - Unicamp, 2001.

[10] L. N. De Castro and J. Timmis. An artificial immune network for multimodal function optimization. *IEEE Congress on Evolutionary Computation (CECÓ2)*, 1:699–674, 2002.

[11] L. N. De Castro and F. J. Von Zuben. Learning and optimization using the clonal selection principle. *IEEE Transactions on Evolutionary Computation*, 6:239–251, 2002.

[12] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[13] P. A. V. Ferreira. Otimização multiobjetivo: Teoria e aplicações. tese de livre docência., 1999.

[14] J. H. Friedman. An overview of predictive learning and function approximation. *NATO ASI SERIES F COMPUTER AND SYSTEMS SCIENCES*, 136:1–1, 1994.

[15] S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Comput.*, 4(1):1–58, 1992.

[16] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.

[17] F. G. Guimarães. Algoritmos de evolução diferencial para otimização e aprendizado de máquina, 2009.

[18] I. Guyon and A. Elisseeff. *An Introduction to Feature Extraction*. Springer, 2006.

[19] S. Haykin. *Redes Neurais: Princípios e Prática*. Bookman, 2001.

[20] G. E. Hinton. Connectionist learning procedures. *Artificial intelligence*, 40(1-3):185–234, 1989.

[21] E. G. Horta, A. P. Braga, and R. R. Saldanha. Acelerando o treinamento multiobjetivo de rnas pelo método de gradiente projetado. In *Congresso Brasileiro de Automática, Setembro*, 2008.

[22] I. Kokshenev and A. P. Braga. A multi-objective approach to rbf network learning. *Neurocomputing*, 71(7):1203–1209, 2008.

[23] P. Korhonen. Multiple objective programming support., 1998.

[24] A. S. M. Lacerda. Proposta de um algoritmo evolucionário nebuloso para solução de problemas de otimização multiobjetivo. Dissertação de mestrado, Universidade Federal de Minas Gerais - UFMG, 2010.

[25] D. Li. Zero duality gap for a class of nonconvex optimization problems. *Journal of Optimization Theory and Applications*, 85(2):309–324, 1995.

[26] L. Liang, G. Xu, D. Liu, and S. Zhau. Immune clonal selection optimization method with mixed mutation strategies. *Second International Conference on Bio-Inspired: Theories and Applications, BIC-TA*, pages 37–41, 2007.

[27] A. Osyczka. Multicriteria optimization for engineering design. *Design optimization*, 1:193–227, 1985.

[28] V. Pareto. Cours d\'economie politique. 1897.

[29] R. H. C. Takahashi R. A. Teixeira, A. P. Braga and R. R. Saldanha. Improving generalization of mlps with multi-objective optimization. *Neurocomputing*, 35(1–4):189–194, 2000.

[30] R. Reed. Pruning algorithms-a survey. *Neural Networks, IEEE Transactions on*, 4(5):740–747, 1993.

[31] H. P. Rocha, C. L. Castro, and A.P. Braga. Seleção de modelos neurais utilizando evolução diferencial através do controle de erro e norma do vetor de pesos. In *X Congresso Brasileiro de Inteligência Computacional (CBIĆ2011)*, November 2011.

[32] D. E. Rumelhart and J. L. McClelland. *Parallel Distributed Processing, vol1: Foundations.* The MIT Press, 1986.

[33] Y. Saeys, I. Inza, and P. Larrañaga. A review of feature selection techiniques in bioinformatics. *Bioinformatics*, 23:2507–2517, 2007.

[34] Y. Sawaragi, H. Nakayama, and T. Tanino. *Theory of multiobjective optimization.* Academic Press, 1985.

[35] M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 111–147, 1974.

[36] R. M. Storn and K. V. Price. Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. *Technical Report TR-95-012, International Computer Science Institute*, page 22, 1995.

[37] F. Tan, X. Fu, Y. Zhang, and A.G. Bourgeois. Improving feature subset selection using a genetic algorithm for microarray gene expression data. *IEEE Congress on Evolutionary Computation*, pages 16–21, 2006.

[38] R. A. Teixeira. *Treinamento de Redes Neurais Artificiais Através de Otimização Multi-Objetivo: Uma Nova Abordagem para o Equilíbrio entre a Polarização e a Variância*. Tese de doutorado, Universidade Federal de Minas Gerais - UFMG, 2001.