Pós-Graduação em Ciência da Computação

“Otimização Evolucionária Multimodal de Redes Neurais Artificiais com Evolução Diferencial”

Por

Antonio Miguel Faustini Zarth

Dissertação de Mestrado

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE, MAIO/2010

UNIVERSIDADE FEDERAL DE PERNAMBUCO

CENTRO DE INFORMÁTICA

PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ANTONIO MIGUEL FAUSTINI ZARTH

"OTIMIZAÇÃO EVOLUCIONÁRIA MULTIMODAL DE REDES NEURAIS ARTIFICIAIS COM EVOLUÇÃO DIFERENCIAL"

ESTE TRABALHO FOI APRESENTADO À PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO DO CENTRO DE INFORMÁTICA DA UNIVERSIDADE FEDERAL DE PERNAMBUCO COMO REQUISITO PARCIAL PARA OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIA DA COMPUTAÇÃO.

ORIENTADOR(A): Teresa Bernarda Ludermir

RECIFE, MAIO/2010

Zarth, Antonio Miguel Faustini

Otimização evolucionária multimodal de redes neurais artificiais com evolução diferencial / Antonio Miguel Faustini Zarth. - Recife: O Autor, 2010.

x, 71 folhas : il., fig., tab.

Dissertação (mestrado) – Universidade Federal de Pernambuco. CIn. Ciência da Computação, 2010.

Inclui bibliografia.

1. Inteligência artificial. 2. Inteligência computacional. 3. Sistemas híbridos. 4. Otimização de redes neurais artificiais. I. Título.

006.3 CDD (22. ed.) MEI2010 – 067

Dissertação de Mestrado apresentada por **Antônio Miguel Faustini Zarth** à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título **"Otimização Evolucionária Multimodal de Redes Neurais Artificiais com Evolução Diferencial"**, orientada pela **Profa. Teresa Bernarda Ludermir** e aprovada pela Banca Examinadora formada pelos professores:

Prof. Cleber Zanchettin
Centro de Informática / UFPE

Prof. Mêuser Jorge Silva Valença
Departamento de Sistemas e Computação / UPE

Profa. Teresa Bernarda Ludermir
Centro de Informática / UFPE

Visto e permitida a impressão.
Recife, 01 de março de 2010.

Prof. Nelson Souto Rosa
Coordenador da Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.

Para minha família.

Agradecimentos

Previamente, meus sinceros agradecimentos à minha orientadora Teresa pela oportunidade oferecida. Também não posso deixar de agradecer pela paciência, bons conselhos e o acompanhamento onipresente que permitiu a realização deste trabalho.

Agradeço à minha família e aos meus amigos que contribuíram e participaram de todos momentos desta jornada: minha namorada Taís pela paciência e apoio, meus pais pelo incentivo incondicional e meu irmão Fernando pelas infindáveis conversas e discussões.

Resumo

Este trabalho propõe uma metodologia de otimização multimodal e simultânea de pesos e arquiteturas de Redes Neurais Artificiais (RNAs) com Evolução Diferencial. O sistema neural híbrido proposto busca por arquiteturas de forma construtiva e realiza o ajuste dos pesos invasivamente, dispensando assim o uso de algoritmos de treinamento por correção de erros. A motivação para o desenvolvimento do presente trabalho é propor uma abordagem que contorne a sensibilidade natural dos métodos construtivos e invasivos aos mínimos locais , tanto na busca de arquiteturas, quanto no ajuste dos pesos das RNAs. Para isto, utilizou-se uma estratégia implícita de manutenção da diversidade inspirada na computação evolucionária paralela. A combinação desta estratégia em um sistema neural híbrido, assim como sua adaptação em uma metodologia construtiva e invasiva, é a principal inovação deste trabalho.

Como base evolucionária da metodologia proposta, foram utilizadas a Evolução Diferencial em sua forma original e também uma recente variação deste algoritmo, a Evolução Diferencial baseada em Oposição. Desta forma, esta dissertação possui dois objetivos primários: (1) avaliar a performance da metodologia proposta comparando com outros sistemas neurais híbridos encontrados na literatura; (2) avaliar e comparar o desempenho dos dois algoritmos evolucionários utilizados na otimização de redes neurais.

Os experimentos foram conduzidos com o propósito de otimizar redes *Multi-Layer Perceptron* (MLP) para problemas de classificação. Os critérios utilizados para análise de performance do método foram a capacidade de generalização, tamanho da arquitetura encontrada e tempo de convergência. Os resultados obtidos indicam que o método proposto possui grande capacidade de generalização com qualidade de resposta superior ou equivalente a muitos métodos encontrados na literatura, e geralmente com menor arquitetura. Além do mais, a metodologia multimodal proposta obteve estes bons resultados com velocidade plausível, necessitando de poucos segundos para convergir. Estas análises ressaltam a boa performance geral do sistema neural híbrido proposto, cuja característica uni-modular sugere que bons resultados podem ser obtidos sem excessiva complexidade e em tempo hábil.

Palavras-chave: Sistemas Híbridos Inteligentes, Redes Neurais Artificiais, Evolução Diferencial, Otimização de Pesos e Arquitetura.

Abstract

This works introduces a multimodal methodology for simultaneous optimization of weights and architectures of Artificial Neural Networks (ANNs) with Differential Evolution. The proposed hybrid neural system searches for architectures in a constructive way and makes the adjustment of weights invasively, avoiding the use of error correction-based training algorithms. The motivation for the development of this work is to propose an approach that bypasses the natural sensitivity of the constructive and invasive methods to the local minimum, both in search for architectures, as in the adjustment of the weights of RNAs. For such, we used an implicit strategy of maintaining diversity inspired by the parallel evolutionary computation. The combination of this strategy in a hybrid neural system, as well as their adaptation in a constructive and invasive method, is the main innovation of this work.

As evolutionary basis of the proposed methodology, we used the Differential Evolution in its original form and also a recent variation of this algorithm, the Opposition-based Differential Evolution. Thus, this dissertation has two primary goals: (1) evaluate the performance of the proposed method compared with others hybrid neural systems found in the literature, (2) evaluate and compare the performance of the two evolutionary algorithms used in the optimization of neural networks.

The experiments were conducted with the purpose of optimizing networks Multi-Layer Perceptron (MLP) for classification problems. The criteria used to analyze the performance of the method are the ability to generalize, size of the architecture found and convergence speed. The results indicate that the proposed method has great ability to generalize with the quality of response better or equal to many methods in the literature, and generally with smaller architecture. Still, the multimodal methodology proposed obtained these good results with reasonable speed, requiring only few seconds to converge. These results highlight the good overall performance of proposed hybrid neural system, whose unimodular characteristic suggests that good results can be obtained without excessive complexity and in a timely manner.

Keywords: Hybrid Intelligent Systems, Artificial Neural Networks, Differential Evolution, Optimization of Weights and Architecture.

Sumário

1 Introdução **1**
- 1.1 Motivação 1
- 1.2 Objetivo deste Trabalho 3
- 1.3 Organização da Dissertação 5

2 Sistemas Neurais Híbridos **7**
- 2.1 Redes Neurais Artificiais 7
  - 2.1.1 Introdução 7
  - 2.1.2 Perceptron Multi-Camadas 8
  - 2.1.3 Treinamento 9
  - 2.1.4 Mínimos Locais 12
- 2.2 Otimização Evolucionária de Redes Neurais Artificiais 13
  - 2.2.1 Introdução 13
  - 2.2.2 Principais Abordagens de SNHs evolucionários 14
- 2.3 Comentários Finais 19

3 Evolução Diferencial **20**
- 3.1 Evolução Diferencial 20
  - 3.1.1 Introdução 20
  - 3.1.2 Mutação 20
  - 3.1.3 Cruzamento 22
  - 3.1.4 Seleção 22
- 3.2 Evolução Diferencial Baseada em Oposição 24
  - 3.2.1 Introdução 24
- 3.3 Otimização de Redes Neurais com Evolução Diferencial 25
- 3.4 Comentários Finais 28

4 Metodologia de Otimização Multimodal de Redes Neurais **30**
- 4.1 Introdução 30
- 4.2 Representação 32
- 4.3 Migração 33
- 4.4 Crescimento da Rede 35
- 4.5 Critério de Parada 36
- 4.6 Funções de Custo 38
- 4.7 Comentários Finais 40

5 Experimentos **42**
5.1 Introdução 42
5.2 Domínios de Problemas 42
5.3 Configuração Geral dos Experimentos 42
5.4 Influência da taxa de mutação no treinamento de RNAs 44
5.5 Capacidade de Generalização e Complexidade da Arquitetura 48
5.5.1 Resultados para o problema do Câncer 49
5.5.2 Resultados para o problema Diabetes 54
5.5.3 Resultados para o problema Íris 56
5.5.4 Resultados para o problema Tireoide 58
5.6 Comparação com Busca Manual de Arquiteturas 60
5.7 Comentários Finais 62

6 Conclusão **64**
6.1 Considerações Finais 64
6.2 Trabalhos Futuros 65

Lista de Figuras

2.1 Arquitetura de um perceptron multi-camadas com duas camadas ocultas. 9
2.2 *Overfitting* no treinamento de RNAs. 11
2.3 Exemplo de mínimos locais em uma superfície de erro. 12
2.4 Problema dos mínimos locais em abordagens construtivas. 16
2.5 Exemplo de um modelo hierárquico de otimização de RNAs. 18

3.1 Mutação com DE/rand/1/bin. 21

4.1 Representação da população. 33
4.2 Migração unidirecional de $S \in 1,2,...,k$ subpopulações. 34
4.3 Representação de um indivíduo da população. 35

5.1 Influência da taxa de mutação no treinamento de RNAs. 46
5.2 Evolução do treinamento em diferentes taxas de F. 47

Lista de Tabelas

4.1 Configurações utilizadas na metodologia multimodal proposta. 41

5.1 Descrição dos domínio de problemas utilizados nos experimentos. 43
5.2 Influência da taxa de mutação no treinamento de RNAs. 45
5.3 Resultados para o problema Câncer 1. 49
5.4 Analise estatística dos resultados do problema Câncer 1. 50
5.5 Resultados para o problema Câncer 2. 51
5.6 Analise estatística dos resultados do problema Câncer 2. 52
5.7 Resultados para o problema Câncer 3. 52
5.8 Analise estatística dos resultados do problema Câncer 3. 53
5.9 Comparação com outros resultados reportados para o problema do Câncer. 53
5.10 Resultados para o problema Diabetes. 54
5.11 Analise estatística dos resultados do problema Diabetes. 55
5.12 Comparação com outros resultados reportados para o problema Diabetes. 55
5.13 Resultados para o problema Iris. 56
5.14 Analise estatística dos resultados do problema Iris. 57
5.15 Comparação com outros resultados reportados para o problema Íris. 57
5.16 Resultados para o problema Tireoide. 58
5.17 Analise estatística dos resultados do problema Tireoide. 59
5.18 Comparação com outros resultados reportados para o problema Tireoide. 59
5.19 Comparação com busca manual de arquiteturas. 61

CAPÍTULO 1

Introdução

1.1 Motivação

Atualmente, Redes Neurais Artificiais (RNAs) formam um paradigma bem estabelecido no campo da inteligência computacional. Devido a sua capacidade de aprender através de exemplos, generalização e tolerância a falhas, as RNA têm sido aplicadas com sucesso em diversos tipos de problemas, tais como os de previsão de séries temporais, classificação, reconhecimento de faces, análise de crédito e controle. Dentre diversos modelos conexionistas com características e topologias diferentes, a RNA Multi-Layer Perceptron [RHW86] treinada com o algoritmo *backpropagation* é uma das mais difundidas, tanto em aplicações práticas quanto em pesquisas diversas. Embora a reconhecida aceitação deste algoritmo, novas propostas tem atraído muita atenção na comunidade científica, visando RNAs mais robustas e precisas.

Este crescente interesse por redes neurais mais eficazes se deve principalmente por suas persistentes limitações. Algoritmos de treinamento simples, fáceis de implementar e de ampla aceitação possuem severas desvantagens que restringem a qualidade de resposta de uma RNA objetivo. Não obstante, um projeto consistente de RNA para a resolução de um problema específico depende de outras variáveis, onde muitas vezes há pouca ou nenhuma informação que possa guiar sua configuração correta. De fato, uma variável crítica refere-se à seleção apropriada da arquitetura da RNA, cuja carência de metodologias capazes de orientar sua definição correta promove um tedioso trabalho de tentativa e erro buscando resultados mais robustos. Logo, a expectativa de sucesso com o uso desta poderosa ferramenta torna-se na maioria das vezes um mero devaneio com objetivos frustrados.

Teoriza-se por uma rede neural ideal, aquela que é capaz de resolver um problema ao qual foi concebida através de uma rápida convergência, arquitetura reduzida e boa capacidade de generalização [Abr04]. Atingir o propósito deste enunciado através da definição empírica dos parâmetros de uma RNA é uma tarefa muito difícil. Considerando que um projeto de RNA é desenvolvido para a resolução de um problema específico, a arquitetura e estratégia de treinamento deve condizer com suas características e propriedades. Para tal, seria necessário um conhecimento a *priori* detalhado sobre o domínio do problema, permitindo em um cenário

ideal, a compreensão de todas suas nuances e o resultado impactante das diversas configurações possíveis.

O uso de otimizadores evolucionários têm se mostrado uma alternativa viável e eficaz para contornar estas limitações e aprimorar a qualidade de resposta dos projetos de RNAs. A otimização evolucionária pode realizar com sucesso essa tarefa, permitindo o ajuste de vários, senão todos, parâmetros livres de uma RNA, buscando soluções precisas. Esta perspectiva se deve à característica distinta dos algoritmos evolucionários, que são capazes de se adaptar a ambientes dinâmicos com grande eficiência [Yao99]. A hibridização, em sua forma geral, busca a combinação de técnicas que possam levar a uma solução mais robusta e eficiente [BCL07]. No entanto, a hibridização de otimizadores evolucionários com RNAs pode combinar suas vantagens, entretanto, também pode unir suas limitações. Devido justamente a sua natureza evolucionária, estes algoritmos são lentos e potencialmente demandam alto custo computacional. Para otimizar RNAs através da busca evolucionária, o sistema híbrido precisa ser equilibrado e bem justificado, mantendo a coerência com o enunciado de RNAs ideais, citado anteriormente.

Enquanto a maioria dos trabalhos encontrados no meio científico propõe Sistemas Neurais Híbridos (SNHs) com o uso de algoritmos evolucionários tradicionais, como o algoritmo genético, pesquisas recentes revelam um crescente interesse na hibridização de RNAs com evolução diferencial. A evolução diferencial (DE, acrônimo de *Differential Evolution*), é um dos mais novos membros da família dos algoritmos evolucionários, sendo proposta em 1995 por Storn e Price [SP95]. Com bons resultados apresentados em diversos estudos com foco em otimização numérica, pesquisas recentes buscaram investigar sua aplicação na área dos sistemas neurais híbridos, mas não raramente com resultados ambíguos em profundidade e definição. Grande parte destas pesquisas limitam-se em aplicar o DE apenas para treinamento ou apenas para o ajuste da arquitetura de redes neurais. Desta forma, uma metodologia que utilize e investigue seus potenciais benefícios na otimização de pesos e arquiteturas de RNAs em único arcabouço é uma lacuna a ser preenchida. Novas propostas híbridas com o uso de DE é uma necessidade óbvia, dada a grande quantidade de trabalhos já publicados com outras técnicas evolucionárias.

O interesse desta dissertação concerne ao desenvolvimento de um SNH híbrido para ajuste de pesos e arquiteturas com o uso do DE, de forma invasiva. De modo geral, SNHs invasivos caracterizam-se pelo objetivo de contornar as limitações dos algoritmos de treinamento como o *backpropagation* através do ajuste de pesos pelo próprio método evolucionário [PHU05]. Entretanto, devido a carência de trabalhos que detalham o treinamento de RNAs com DE, a configuração paramétrica dos seus operadores de variação baseia-se nas recomendações dos trabalhos não voltados a sistemas híbridos. Uma destas especificações sugere como tamanho da população a ordem de dez vezes a dimensionalidade do problema para evitar perda de

diversidade, o que seria impraticável em um problema de alta dimensionalidade como o de treinamento de RNAs. Logo, novas estratégias são necessárias para lidar com esta problemática.

Por fim, mas não menos importante, o ajuste de arquiteturas de forma eficaz depende de uma estratégia eficiente no algoritmo híbrido. Redes muito grandes podem perder sua capacidade de generalização por ajuste excessivo de pesos, além do óbvio custo computacional. Por outro lado, arquiteturas inadequadamente pequenas podem não ter complexidade suficiente para tratar o problema para o qual é concebida. Uma estratégia bastante difundida é de otimizar arquiteturas de RNAs através de algoritmos construtivos, com o propósito de garantir redes pequenas. Entretanto, esta técnica incremental limita a exploração do espaço de busca por arquiteturas, muitas vezes encontrando mínimos locais e interrompendo o algoritmo precocemente. Dentre os diversos modelos propostos na literatura, algoritmos invasivos e construtivos mostram-se promissores na área dos sistemas neurais híbridos. Mas, como em qualquer metodologia, nesta classe de SNH permanecem desafios cabíveis de solução, incentivando novas abordagens.

A otimização evolucionária de redes neurais mostra-se uma área de pesquisa moderna e promissora, entretanto, percebe-se que outras limitações surgem neste escopo. Se a otimização evolucionária de redes neurais tem como proposta resolver problemas de alta complexidade em busca de resultados melhores e mais refinados, é importante que esta não fique inserida nas limitações dos algoritmos polimórficos que o sistema híbrido propõe. Diante desta retórica, a necessidade de avançar as pesquisas nesta área, investigando e propondo arcabouços que possam levar a sistemas híbridos mais robustos, eficientes e precisos, é cada vez mais evidente.

1.2 Objetivo deste Trabalho

O foco desta proposta está concentrado no desenvolvimento de uma metodologia construtiva e invasiva de otimização simultânea de pesos e arquiteturas com evolução diferencial. O objetivo principal é propor um sistema neural híbrido evolucionário com características multimodais na seleção construtiva de arquiteturas e treinamento de RNAs. Esta metodologia é fundamentada com base nas diversas propostas de SNHs da literatura, levando em consideração suas vantagens e limitações. Desse modo, o trabalho apresenta de forma distinta uma adaptação de uma estratégia implícita de manutenção da diversidade em otimizadores baseados em população na aplicação de um sistema neural híbrido. Contudo, a principal contribuição deste trabalho remete às funções projetadas e delegadas ao modelo multimodal: manutenção da diversidade e seleção multimodal de arquiteturas e pesos.

O conceito de multimodal é definido como um problema onde existem pontos no espaço de busca que são melhores que seus vizinhos mas não são ótimos, ou seja, não propõem uma solução tão boa como a melhor solução global [ES03]. Em problemas complexos e não lineares, é normal que o espaço de busca possua diversas bacias de atração, também chamadas de ótimos locais. Este ótimos locais representam a melhor solução de uma região delimitada do espaço de busca, entretanto, é possível que não seja a melhor possível. Encontrar a melhor solução global sem ficar preso em ótimos locais é um dos maiores desafios dos algoritmos de busca, otimização, ou ainda no treinamento de redes neurais.

Embora geralmente consigam redes menores que algoritmos baseados em poda [yKY97], algoritmos construtivos tendem a ficar presos em mínimos locais devido as propriedades multimodais e deceptivas das redes neurais [ASP94]. Esta limitação esta diretamente relacionada com sua natureza incremental. Assim, a metodologia proposta aborda uma estratégia evolucionária construtiva que avalia diversas arquiteturas de forma não exaustiva ou hierárquica, com o propósito de buscar boas soluções antes de julgar necessária a interrupção da busca.

Da mesma forma, a metodologia multimodal utiliza esta estratégia para manutenção da diversidade e melhor ajuste de pesos (treinamento) da RNA, mantendo a dependência entre pesos com respectivas arquiteturas testadas. A partir do pressuposto que manter a diversidade implica em uma convergência melhor evitando a estagnação evolucionária, o modelo proposto também pretende realizar a busca por pesos e arquiteturas em tempo hábil se comparado com outros métodos, utilizando uma população de tamanho reduzido e usufruindo das vantagens do processamento paralelo.

Para tal proposta, uma análise das principais características e limitações do desenvolvimento de um projeto de RNA são discutidas, servindo como base teórica para uma melhor compreensão dos desafios e objetivos dos sistemas neurais híbridos. A partir desta conceituализação, um estudo foi realizado sobre as principais metodologias de SNHs evolucionários, apresentadas e exemplificadas com algumas pesquisas relevantes. Adicionalmente, utilizamos uma importante variação do DE proposta recentemente: a evolução diferencial baseada em oposição (ODE) [RTS08]. De acordo com a revisão da literatura feita para esta dissertação, não há relatos de seu uso como otimizador de redes neurais, sendo esta uma das contribuições deste trabalho.

Além de propor uma metodologia construtiva e invasiva de otimização simultânea de pesos e arquiteturas com características multimodais, espera-se que os resultados e referenciais teóricos devidamente discutidos neste trabalho possam servir como material útil, guiando novas propostas e contribuições na área dos sistemas híbridos.

1.3 Organização da Dissertação

Esta dissertação está organizada em seis capítulos, incluindo esta introdução que apresentou a motivação e objetivos gerais do trabalho.

CAPÍTULO 2: Sistemas Neurais Híbridos

Este capítulo é divido em duas partes co-relacionadas e de igual importância. A primeira parte discute os conceitos básicos de redes neurais artificiais, com foco nas redes do tipo MLP. Estes conceitos são apresentados dando ênfase nas limitações e desafios em detrimento do desenvolvimento de um projeto eficaz. A partir destas considerações, a segunda parte do capítulo apresenta o conceito de sistemas neurais híbridos, com foco na otimização evolucionária de RNAs. Também é apresentada e caracterizada uma taxonomia geral das principais abordagens, incluindo suas vantagens, deficiências e alguns trabalhos relevantes.

CAPÍTULO 3: Evolução Diferencial

O capítulo três explica em detalhes o método evolucionário utilizado no desenvolvimento desta proposta: a Evolução Diferencial. Adicionalmente, é introduzida uma variação recente deste algoritmo, a Evolução Diferencial Baseada em Oposição. O capítulo encerra com uma breve discussão de trabalhos na área de sistemas neurais híbridos que utilizam esta técnica evolucionária, complementarmente à revisão da literatura do capítulo anterior.

CAPÍTULO 4: Metodologia de Otimização Multimodal de Redes Neurais

Este capítulo apresenta o método proposto, objetivo fundamental desta dissertação. É apresentada a metodologia desenvolvida, explicando a integração da Evolução Diferencial em um ambiente paralelo de otimização simultânea de pesos e arquiteturas de RNAs. Ainda neste capítulo são apresentados os detalhes de funcionamento bem como reafirmados os objetivos e justificativas que fundamentam seu desenvolvimento.

CAPÍTULO 5: Experimentos

Para validar a metodologia proposta, experimentos foram realizados e descritos no capítulo 5, destacando as estratégias de execução aplicadas. São apresentados os domínios de proble-

mas utilizados, os diversos experimentos realizados e respectivos resultados obtidos.

CAPÍTULO 6: Conclusão

O capítulo que encerra esta dissertação apresenta as considerações finais, discutindo de forma breve as características, inovações, vantagens e limitações relevantes desta dissertação. Por fim, o capítulo de conclusão encerra com uma lista de propostas de trabalhos futuros, objetivando dar continuidade ao trabalho.

CAPÍTULO 2

Sistemas Neurais Híbridos

2.1 Redes Neurais Artificiais

2.1.1 Introdução

Redes neurais artificiais (RNAs), ou sistemas conexionistas, buscam representar de alguma forma a estrutura do cérebro humano para resolver problemas de inteligência computacional. Mais especificamente, os modelos conexionistas caracterizam-se pela reunião de várias unidades de processamento interligadas por conexões que processam paralelamente a informação [Hay98], com desempenho e características semelhantes as redes neurais biológicas. Apesar dos primeiros estudos desenvolvidos datarem de 1943 [MP43] e a primeira aplicação prática de um modelo conexionista datar de 1958 [Ros58], apenas nas últimas décadas a área de pesquisa em redes neurais vêm despertando grande interesse e se expandindo rapidamente [NGN05]. Isto se deve por diversos fatores, como por exemplo, melhores conhecimentos da estrutura do cérebro, melhores algoritmos de treinamento e disponibilidade de computadores poderosos [Bit06], permitindo que as RNAs sejam aplicadas com sucesso em inúmeras áreas de conhecimento, particularmente predição, classificação, reconhecimento e controle.

Usualmente, uma RNA é constituída por unidade de processamento simples (também chamados de neurônios ou nodos), que processam determinadas funções matemáticas, geralmente não-lineares [Hay98]. Os nodos são distribuídos em camadas e geram um valor de saída de acordo com sua função de ativação e sinal de entrada, interagindo com outros neurônios ao enviar suas saídas através de conexões. Estes nodos são ponderados pelas conexões ao qual estão associados, podendo alterar ou não o sinal que transmitem. Nos modelos tradicionais de RNAs, estas conexões, também chamadas de pesos, são responsáveis por armazenar o conhecimento representado no modelo.

A larga utilização das RNAs deve-se principalmente pela sua capacidade de aprendizado. Na maioria dos casos, uma RNA é um sistema adaptativo que muda sua estrutura baseada nas informações apresentadas, dando-lhe a capacidade de aprender através de exemplos. Através do aprendizado, RNAs podem generalizar o conhecimento adquirido, ou seja, após o treinamento,

espera-se que a rede possa reconhecer instâncias similares que não haviam sido apresentadas anteriormente [BCL07, Hay98]. Em outras palavras, RNAs conseguem modificar suas conexões sinápticas para fazer um mapeamento entrada-saída com um conjunto reduzido de dados de treinamento, permitindo posteriormente dar respostas coerentes para dados não conhecidos.

Entre outras características, as RNAs também destacam-se pelo bom desempenho e tolerância a falhas. Quando a rede encontra dados faltantes ou imprecisos, geralmente esta exibe uma degradação suave do desempenho em vez de apresentar uma falha catastrófica. Juntamente com a habilidade de aprendizado e a estrutura maciçamente paralela, estas propriedades tornam as RNA's ferramentas computacionais poderosas, aptas a resolver problemas de grande escala. [Bit06, BCL07, Hay98]

Como exemplificado anteriormente, o grande interesse da comunidade científica em rede neurais se deve aos vários casos de sucesso em diversas aplicações. Apesar destas distintas vantagens e ampla utilização em uma infinidade de aplicações, os problemas do mundo real mostram-se dinâmicos e não raramente muito complexos para serem resolvidos de forma trivial com o uso de redes neurais artificiais. A eficiência de uma RNA depende principalmente da escolha específica de seus parâmetros que precisam contribuir tanto para o bom desempenho da rede como para a definição de uma rede simples, com as especificações adequadas para solucionar o problema para o qual foi definida [BCL07]. Para isto, uma rede neural hipoteticamente ótima para um determinado problema deve ter como características principais possuir uma arquitetura de baixa complexidade e ser treinada rapidamente, mas que permita boa performance de generalização [Abr04, Yao99].

2.1.2 Perceptron Multi-Camadas

A arquitetura de uma rede neural é determinada pela sua estrutura topológica, ou seja, sua conectividade, e pela função de transferência (ativação) de cada neurônio da rede [Yao99]. Dentre as diversas topologias, a RNA mais difundida é a perceptron multi-camadas, mais conhecida por MLP (do inglês Multi Layer Perceptron) [RHW86]. O sucesso e ampla aceitação da rede MLP deve-se principalmente ao fato de resolverem problemas não linearmente separáveis, simplicidade e facilidade de implementação [Hay98].

Tipicamente, uma RNA do tipo MLP caracteriza-se por uma arquitetura altamente conectada disposta em camadas, sendo uma de entrada, uma ou mais camadas intermediárias e uma camada de saída, como mostra a figura 2.1. Com exceção dos nodos da camada de entrada, todos os demais neurônios possuem capacidade computacional. O número de neurônios da camada de entrada depende da dimensão do espaço de observação que gera os sinais de entrada,

ou seja, o número de variáveis do problema. Pelo fato de não serem nós computáveis, sua função é propagar as variáveis de entrada através das conexões para a camada intermediária.

As camadas intermediárias, também chamadas de camadas ocultas, são responsáveis pela não linearidade da rede. Os neurônios desta camada possuem uma função de ativação não-linear, geralmente a função sigmoidal tangente hiperbólica ou logística. A camada intermediária funciona como um detector de características. Com um número suficiente de neurônios nesta camada, é possível gerar uma codificação interna para qualquer padrão de entrada [BCL07]. Não existe uma notação formal para especificar o número de camadas intermediárias, porém de acordo com [BCL07], uma camada é suficiente para aproximar qualquer função contígua, sendo apta para a maioria dos problemas. Da mesma forma, não há uma regra para a quantidade de nodos na camada oculta, que é geralmente definida empiricamente.

Por fim, a última camada de nodos da rede constitui a camada de saída, cujo sinal funcional será a resposta dada pela rede dos dados de entrada fornecidos. Assim como a camada de entrada, a quantidade de neurônios na camada de saída é dependente do problema, ou seja, a quantidade de atributos que a resposta do problema demanda.

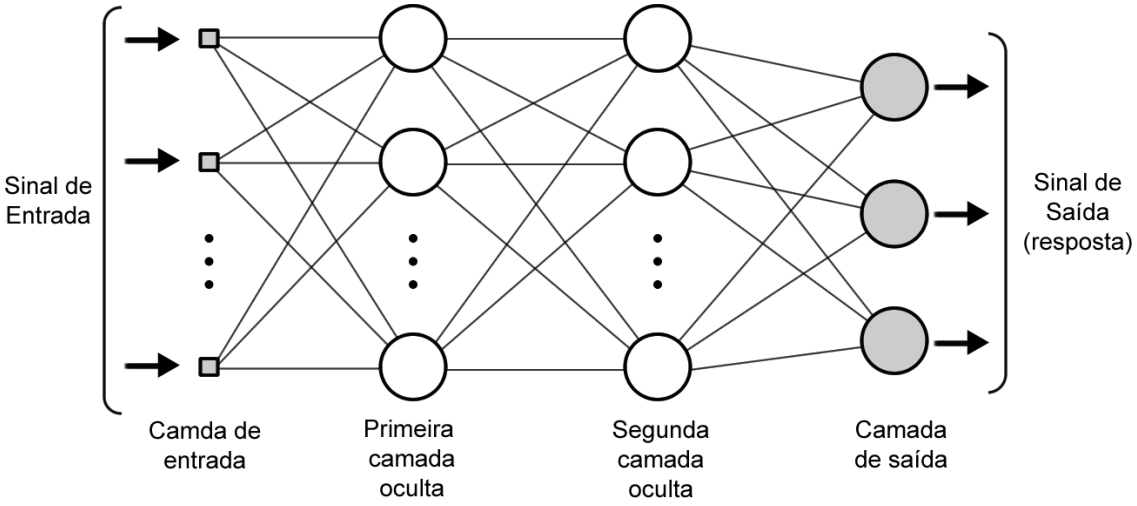

Figura 2.1 Arquitetura de um perceptron multi-camadas com duas camadas ocultas.

2.1.3 Treinamento

Nas redes MLPs, os nodos de uma camada interligam-se aos nodos da camada adjacente através de conexões, onde cada uma possui um peso associado. Este conjunto de pesos é responsável por armazenar o conhecimento representado da rede. Desta forma, ao ajuste deste conjunto

de pesos se dá o nome de treinamento, cujo objetivo é adaptar esta estrutura interna da RNA de forma a corresponder a saída desejada do problema. Em um sentido implícito, o processo de aprendizagem deve decidir quais características do padrão de entrada devem ser representadas [Hay98]. Existem várias técnicas e algoritmos de aprendizado de redes MLPs, diferindo principalmente na forma como os pesos são modificados.

Para treinar uma rede do tipo MLP o algoritmo utilizado mais conhecido e utilizado é a retro propagação de erro, mais conhecido por *backpropagation* [RHW86]. Durante o treinamento, os padrões de entrada são propagados através da rede, camada por camada, gerando um erro que é calculado pela diferença da saída da rede com a saída desejada. De forma geral, o *backpropagation* propaga esse erro para trás, começando pela camada de saída, ajustando os pesos sinápticos através de uma taxa de aprendizado de modo a fazer com que a resposta real da rede se mova para mais perto da resposta desejada [Hay98]. Após um número suficiente de épocas (ciclos), espera-se que o ajuste de pesos leve a rede a um estado cujo erro seja aceitavelmente pequeno para a função objetivo.

Apesar de popular e eficaz na maioria dos problemas, o *backpropagation* e outros algoritmos de treinamento baseados em gradiente possuem algumas reconhecidas desvantagens. Um grande número de interações durante o aprendizado é necessário para obter um resultado satisfatório, demandando tempo e alto custo computacional. Em redes neurais grandes (com muitos neurônios) este problema se acentua, pois a medida que a rede cresce, a dimensão das matrizes derivativas e de covariância de aproximação do erro cresce exponencialmente. Outro problema enfrentado é relacionado à definição correta dos parâmetros. Algoritmos de treinamento para redes neurais baseados em correção por erros são dependentes dos parâmetros de aprendizado. Quando o coeficiente de aprendizado é muito pequeno, a convergência da rede demandará muitas iterações. De outra forma, uma taxa de aprendizado alta poderá tornar a rede instável [BCL07].

Um treinamento bem sucedido não está apenas relacionado com o algoritmo de aprendizado escolhido, mas com outros fatores como: tamanho da rede e critério de parada. Um problema que frequentemente ocorre no treinamento de MLPs é o *overfitting*. Este problema ocorre quando a rede passa a decorar os padrões de entrada durante o treinamento, limitando sua capacidade de generalização [BCL07, Hay98], conforme mostra a figura 2.2. Dado que não existe uma regra que indique a quantidade necessária de neurônios ocultos para resolver cada tipo de problema, a definição correta da arquitetura da RNA é um problema crítico. Uma definição de arquitetura de tamanho exagerado, além de exigir muito tempo de treinamento, pode comprometer completamente a eficiência da RNA, pois esta pode aprender além das características relevantes do conjunto de treinamento, como ruídos implícitos, levando con-

sequentemente ao *overfitting*. Da mesma forma, um número muito pequeno de neurônios pode levar ao problema do *underfitting*. Neste caso, a rede de baixa complexidade pode perder muito tempo tentando encontrar uma representação ótima, levando ao sobrecarregamento dos pesos. Quando o problema é complexo e a rede insuficientemente pequena, a presença de *underfitting* indica que arquitetura não é suficientemente complexa para lidar com todos os nuances dos dados de treinamento, necessários para um aprendizado adequado.

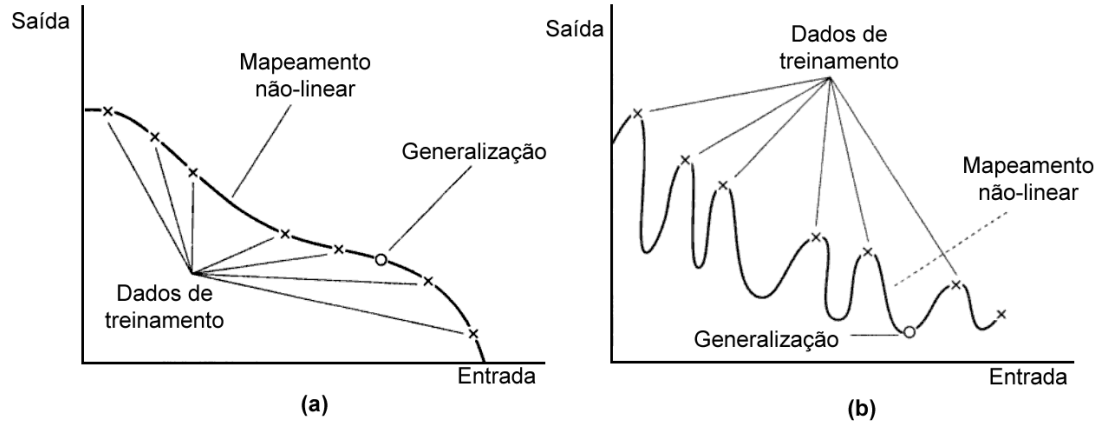

Figura 2.2 *Overfitting* no treinamento de RNAs.
(a) Pesos ajustados adequadamente (boa generalização). (b) *Overfitting* por ajuste excessivo dos pesos (generalização pobre).

Para evitar o problema da perda de generalização por excesso de treino, e consequentemente *overfitting*, uma das formas é estimar a capacidade de generalizar dados desconhecidos durante o treinamento. Para isso, um conjunto de treinamento é utilizado para ajustar os pesos e um conjunto de validação é apresentado para estimar o erro de generalização. Dessa forma, deve-se interromper o treinamento quando este erro estimado do conjunto de validação começar a subir, ou seja, quando a rede começar a aprender além das características relevantes do conjunto de treinamento. Embora essa alternativa se mostre adequada, a própria definição do critério de parada é um passo complexo, já que durante o treinamento o erro estimado de generalização costuma oscilar, não mostrando claramente quando a aprendizagem deve ser interrompida.

Como pode-se perceber nos parágrafos anteriores, o treinamento de redes neurais é uma tarefa complexa que envolve diversos fatores, cujas variáveis críticas referem-se aos problemas intrínsecos do próprio algoritmo de treinamento, definição de uma arquitetura adequada e critério de parada. Configurar corretamente os parâmetros de um projeto de RNA de modo que haja um bom desempenho da rede no problema ao qual ela foi projetada, é muito difícil. Não obstante, estas mesmas variáveis estão relacionadas ao problema dos mínimos locais, cuja

descrição é dada na seção seguinte.

2.1.4 Mínimos Locais

O objetivo do treinamento é encontrar um conjunto de pesos que permita que a saída da RNA corresponda o mais próximo possível a saída desejada. Entretanto, uma boa resposta pode não ser a melhor resposta, pois não são raras as vezes em que os algoritmos de treinamento ficam presos em mínimos locais. Algoritmos mais simples de treinamento baseados em correção de erros, como o *backpropagation*, possuem uma elevada sensibilidade a ficarem presos e interromperem o treinamento antes de encontrarem o mínimo global. [GT92, Hay98, AN00, BCL07, Yao99]. Esta constatação é verificada pela performance do algoritmo ao longo da superfície de erros, que na busca pelo ótimo global pode encontrar diversas bacias de atração, como exemplificado na figura 2.3. No aprendizado por retro-propagação de erros, o problema com mínimos locais é mais perceptível, dado que o algoritmo age como uma "subida da encosta" ficando preso quando a o erro começa a subir, antes de encontrar o mínimo global.

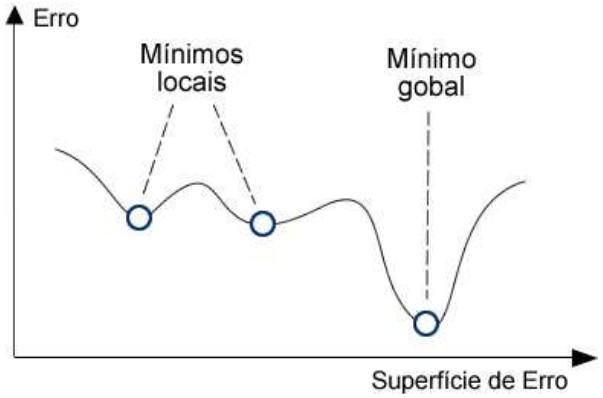

Figura 2.3 Exemplo de mínimos locais em uma superfície de erro.

Ainda neste contexto, é importante destacar que o problema dos mínimos locais, em um âmbito global, está além dos ajuste de pesos. Como explicado anteriormente, uma RNA ótima para um determinado problema é aquela que possui uma arquitetura pequena, com rápida convergência e boa performance de generalização [Abr04]. Entretanto, o espaço de busca de uma RNA, que compreende sua arquitetura e aprendizado, é deceptivo e multimodal [BCL07]: deceptivo pois duas arquiteturas semelhantes podem apresentar desempenhos muito diferentes e multimodal, porque duas arquiteturas muito distintas podem ter desempenhos semelhantes.

Dessa forma, definir uma arquitetura pequena, mas que permita boa generalização, não é uma tarefa simples. Como explicado na seção 2.1.3, uma arquitetura muito pequena não proverá uma generalização suficiente, enquanto uma arquitetura demasiadamente grande demandará excessivo esforço computacional, além de estar suscetível a ruídos da amostra de treinamento. De acordo com [LV91], tentar encontrar uma arquitetura mínima ótima é usualmente um problema NP-difícil. Dessa forma, a definição de uma arquitetura apenas razoável exige um dispendioso e improdutivo trabalho de tentativa e erro.

2.2 Otimização Evolucionária de Redes Neurais Artificiais

2.2.1 Introdução

Desenvolver um projeto de RNAs eficiente, principalmente em problemas complexos, está longe de ser uma tarefa trivial. Como dito anteriormente, o eventual sucesso de um projeto em problemas críticos e que demandam alta precisão, depende muito da experiência do projetista que terá de definir os diversos parâmetros livres de uma RNA. De qualquer forma, existem poucas regras razoavelmente eficientes capazes de guiar o projetista no desenvolvimento de um arcabouço de alto desempenho destinado a resolver um determinado problema.

Neste escopo, o uso de técnicas alternativas para otimizar redes neurais artificiais têm se tornado uma abordagem popular para evitar este trabalho pouco produtivo de tentativa e erro na definição eficiente de seus parâmetros, incentivando novas pesquisas na área de sistemas híbridos como alternativa às técnicas convencionais. Além da busca dos parâmetros livres, sistemas híbridos também podem ser usados com o objetivo de suprir as deficiências das técnicas tradicionais de treinamento de redes neurais, tais como: sensibilidade a mínimos locais, alto custo computacional, necessidade de ajuste de taxas de treinamento, entre outros.

Entende-se por sistemas híbridos a combinação de duas ou mais técnicas que possam levar a uma solução mais robusta e eficiente, suprindo as deficiências e limitações que uma técnica pode apresentar quando aplicada individualmente [BCL07]. Diferente de técnicas de inteligência artificial convencionais que somente trabalham com certeza, precisão e rigor, sistemas híbridos podem explorar a tolerância à imprecisão, incerteza, soluções de baixo custo, robustez, tratamento de verdades parciais e melhor relacionamento com a realidade [AN00]. Nesta grande área, encontram-se os Sistemas Neurais Híbridos (SNHs), que fazem uso de técnicas alternativas para suprir as limitações dos algoritmos de treinamento tradicionais como o backpropagation [RHW86], e/ou otimização paramétrica da rede neural. Existem diversos focos de pesquisa na área de sistemas neurais híbridos, como o de combinar RNAs com técnicas que

se utilizam de conhecimento [BCL07], entretanto esta seção dará ênfase aos sistemas neurais híbridos evolucionários, cujo tópico é de maior relevância aos objetivos desta dissertação.

Os SNHs viabilizam automatizar as tarefas lentas e experimentais no ajuste paramétrico, assim como podem suprir as deficiências dos algoritmos de treinamento convencionais, como os baseados em gradiente. Mas é importante ressaltar que os SNHs podem, mas não necessariamente garantem uma performance geral melhor que técnicas isoladas e/ou tradicionais na otimização de redes neurais. Muitas vezes, técnicas de hibridização podem apresentar exagerada complexidade para uma contribuição muito pequena, impedindo sua aplicação além dos moldes experimentais. Dessa forma, sistemas neurais híbridos mal projetados e pouco justificados podem demandar muito tempo, custo elevado ou ainda resultados poucos confiáveis. Na literatura não são raros os modelos que possuem boa performance para um determinado tipo de problema, e resultados pueris em quaisquer outros. Além disso, sistemas neurais híbridos podem unir e combinar as vantagens das técnicas individuais, mas também seus pontos fracos.

2.2.2 Principais Abordagens de SNHs evolucionários

O uso de algoritmos de otimização baseados em população no treinamento de redes neurais artificiais [Hol75, ML97, IKL03, AK09, SB08] é uma abordagem muito utilizada para evitar as deficiências dos algoritmos de treinamento por retro-propagação de erros [AN00, PHU05]. Abordagens típicas, igualmente, utilizam os benefícios da otimização evolucionária para ajustar a arquitetura da RNA [AN00, ZL07, TSHI07, Bhu09, GTT08], fator crítico para o bom desempenho do modelo conexionista. A otimização evolucionária de redes neurais segue basicamente duas principais abordagens, denominadas no trabalho de Palmes [PHU05] como: invasivas e não-invasivas. As abordagens não-invasivas utilizam otimizadores evolucionários para ajustar a arquitetura da rede, mas dependem de um algoritmo de treinamento por correção de erro para avaliar a aptidão dos indivíduos e realizar o ajuste dos pesos. De outra forma, as técnicas invasivas tentam otimizar a arquitetura e treinar a rede utilizando apenas técnicas evolucionárias.

A partir do pressuposto que algoritmos como o *backpropagation* são amplamente utilizados e estudados, abordagens de SNHs não invasivas são mais populares. Nesta abordagem, a separação explícita das operações entre a adaptação dos pesos por *backpropagation* e da estrutura do processo evolucionário muitas vezes exige o desenvolvimento de um sistema de dupla representação. Sob esta definição, a técnica evolucionária serve apenas como "background"para o treinamento tradicional da RNA com *backpropagation* [PHU05]. Assim, a rede neural na sua forma individual continuará propensa a enfrentar o problema dos mínimos locais deste

algoritmo [GT92].

Abordagens invasivas buscam justamente evitar estes problemas bem conhecidos de técnicas de treinamento como o *backpropagation*, resultando em uma busca mais robusta [PHU05]. Utilizando um sistema evolucionário para todas as tarefas pertinentes à otimização de RNAs, menos parâmetros são necessários para a configuração do SNH e uma representação única pode ser desenvolvida. Como resultado, é possível uma avaliação de aptidão mais rápida e eficaz, tanto para pesos quanto para a arquitetura da RNA.

De formal geral, abordagens invasivas e não invasivas são desenvolvidas sobre três tipo de configurações: metodologias construtivas, baseadas em poda ou ainda hierárquicas. Estas estratégias nem sempre possuem uma fronteira clara dependendo da complexidade do modelo, mas englobam praticamente todos os trabalhos sobre SNHs evolucionários que buscam otimização de arquiteturas e pesos de RNAs. De acordo com o objetivo, estas configurações possuem suas vantagens e desvantagens, cujas descrições e trabalhos relevantes são listados a seguir.

Metodologias construtivas começam com uma RNA com arquitetura mínima, e então novos neurônios na camada oculta são adicionados até uma solução satisfatória ser encontrada [ASP94, PHU05]. A principal vantagem dos algoritmos construtivos é o fato de ser simples especificar a arquitetura inicial, o que não ocorre em métodos de poda da rede. Outra clara vantagem diz respeito ao tamanho da rede solução. Algoritmos construtivos exploram arquiteturas pequenas primeiro, exigindo menos tempo necessário para o treinamento, gerando dessa forma uma economia de recursos computacionais [yKY97].

A vantagem de encontrar redes pequenas em algoritmos construtivos pode ser uma ambígua desvantagem do ponto de vista da arquitetura da RNA. Considerando as características deceptivas das redes neurais, abordagens construtivas tendem a ficar presas em mínimos locais a respeito da relação entre pesos e arquiteturas [ASP94], como exemplifica a figura 2.4. RNAs com arquiteturas distintas podem ter performance semelhantes, assim como arquiteturas semelhantes podem apresentar resultados muito diferentes. A partir deste ponto de vista, a construção incremental dos algoritmos construtivos pode ser interrompida antes de se encontrar uma solução ideal.

Como a adição dinâmica de novos nodos não é uma tarefa complexa, a maior parte dos trabalhos na literatura não utilizam otimizadores para esta função [yKY97]. Utilizando regras algorítmicas para o desenvolvimento da arquitetura, trabalhos recentes como o [IM01] apresentaram bons resultados. Neste trabalho, denominado *Cascade neural networks design algorithm* (CNNDA), os autores seguem basicamente os conceitos fundamentais da evolução por construção, onde inicialmente uma rede mínima é instanciada e então treinada por *backpropagation*. Após esta RNA treinada, o algoritmo descrito em [IM01] adiciona um neurônio por camada

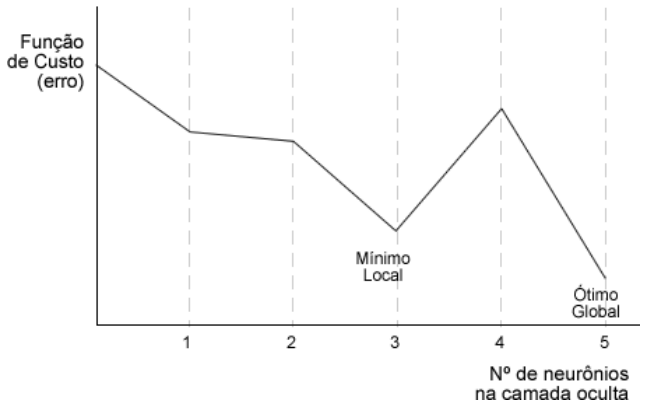

Figura 2.4 Problema dos mínimos locais em abordagens construtivas.

oculta e avalia esta contribuição para a performance da rede. O arquitetura construída pelo CNNDA é disposta em forma de cascata, pois cada neurônio possui conexões com todas camadas adjacentes. Distintamente, o próximo passo do algoritmo é tentar podar conexões da rede após a inserção dinâmica do neurônio. Se a rede perder performance, o algoritmo para. Os resultados expressos neste trabalho apontam bons resultados com convergência rápida, necessitando de poucos segundos para apresentar uma boa solução na maioria dos problemas utilizados. Outros trabalhos igualmente interessantes são descritos em [ASP94, yKY97, MK04] com distintas metodologias construtivas, dada a ampla liberdade de como elaborar regras de criação dinâmica dos novos nodos e da avaliação de performance.

Outro trabalho que utiliza uma metodologia invasiva para otimização simultânea de pesos e arquitetura de RNAs é proposto em [PHU05] e denominado por MGNN: *Mutation-Based Genetic Neural Network*. Neste trabalho a busca por arquiteturas e pesos é realizada utilizando estratégias de mutação gaussiana e também inspiradas na programação evolucionária em duas matrizes conectadas que representam os pesos de uma RNA, e consequentemente sua arquitetura. Uma característica importante deste modelo é que o algoritmo dispensa operações de cruzamento para preservar a dependência entre os pais e a respectiva geração de novos indivíduos. O algoritmo é inicializado com seus indivíduos representados por matrizes de tamanho pré-determinados e com valores zerados. Desta forma, os processos de mutação modificam os valores destas matrizes estabelecendo assim novas sinapses, ou seja, novas conexões não nulas na RNA. Apesar deste comportamento construtivo, MGNN possui representação com tamanho máximo fixo e pré-determinado, delimitando o tamanho máximo da arquitetura. Em relação aos resultados experimentais apontados, o algoritmo é aparentemente simples e de baixo custo computacional, embora não apresente explicitamente dados de tempo ou complexidade que possam reafirmar esta característica. Ainda assim, os experimentos realizados utilizando o mé-

todo MGNN apresentaram soluções com boa performance e arquitetura simples. Entretanto, sua característica mais notável é sua alta confiabilidade, apresentando um baixíssimo desvio padrão como resultado das experimentações

Seguindo o caminho inverso, algoritmos de poda eliminam neurônios e conexões desnecessárias até atingir seu objetivo. Técnicas baseadas em poda possuem a desvantagem da necessidade de se estimar a priori o tamanho mínimo necessário da rede para determinado problema [MK04], o que nem sempre é plausível fazer de forma aproximada. A partir desta arquitetura máxima, é de responsabilidade do processo evolucionário reduzir a arquitetura da RNA enquanto ela não perder performance.

Com esta abordagem, o trabalho [LYZ06] utiliza uma metodologia não invasiva parcial para ajuste da arquitetura e dos pesos de uma RNA, utilizando uma estrutura para representar os pesos e outra para as conexões ativas. Embora faça uso do *backpropagation*, o treinamento final só é realizado com algumas melhores soluções. Para avaliação inicial da aptidão o processo de otimização é utilizado. Metodologias baseadas em poda apresentam bons resultados e são de fácil implementação, mas conforme discutido anteriormente, geralmente não garantem uma arquitetura mínima adequada para o propósito de uma RNA ideal.

O modelo GaTSa proposto por Zanchettin e Ludermir [ZL07] busca otimizar redes neurais combinando *tabu search*, *simulated anneling*, algoritmos genéticos e *backpropagation* para ajuste não invasivo dos pesos. Em relação ao ajuste da arquitetura, o modelo GaTSa faz uso de abordagens construtivas e baseadas em poda de conexões, buscando arquiteturas simples e com alta capacidade de generalização. O algoritmo utiliza algoritmo genético para propor novas soluções e então avaliadas quanto sua capacidade de generalização utilizando *backpropagation*. Do conjunto de soluções, a melhor é escolhida utilizando *tabu search* que será aceita se esta tiver uma função de custo menor que a melhor solução encontrada. Caso esta nova solução não seja a melhor, ela ainda pode ser selecionada dada com uma determinada probabilidade definida pelo *simulated anneling*. O método GaTSa apresentou bons resultados quando comparado com a busca manual de arquiteturas e também em comparação com a utilização das técnicas envolvidas utilizadas de forma independente.

Algoritmos hierárquicos mimetizam, mas automaticamente, o tedioso processo de seleção correta de parâmetros e arquitetura da rede neural. Como já enfatizado, escolher uma arquitetura eficiente manualmente é uma tarefa complexa. Do ponto de vista usual, o projetista treina uma série de redes neurais com tamanhos diferentes e então seleciona uma rede pequena que satisfaça os requerimentos de performance. Da mesma forma, algoritmos baseados em hierarquia podem testar diversas configurações de parâmetros de RNAs até encontrar uma solução adequada, como mostra a figura 2.5.

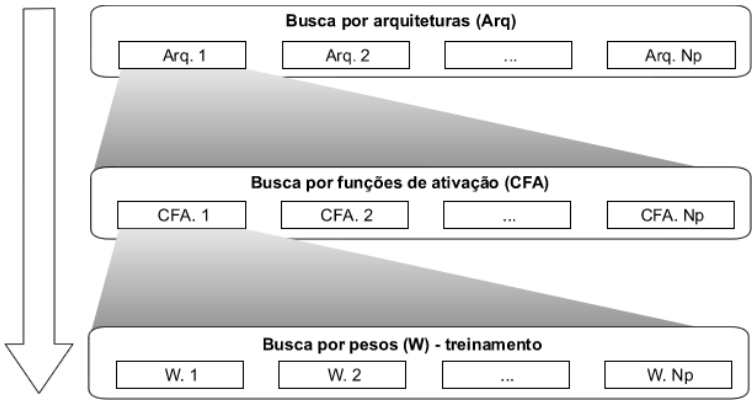

Figura 2.5 Exemplo de um modelo hierárquico de otimização de RNAs.
Busca por arquitetura, funções de ativação e pesos distribuídas em camadas

A grande vantagem dos modelos hierárquicos distribuídos em camadas é sua flexibilidade na escolha da quantidade de parâmetros a serem ajustados. Com um bom modelo, é possível ajustar todos parâmetros livres da RNA, obtendo geralmente redes de alta performance. Contudo, modelos hierárquicos tendem a levar a uma RNA eficiente sob o custo de uma velocidade de convergência lenta. Por utilizar uma evolução recursiva, o modelo demanda muita memória para armazenar as diversas estruturas de representação, exigindo muito tempo para encontrar uma solução. Modelos hierárquicos precisam reiniciar toda a população a medida que a rede cresce, descartando informações relevantes da busca anterior e demandando mais tempo para convergir. Como exemplo, a reinicialização do processo evolucionário de otimização da RNA é realizada quando a matriz de pesos de uma rede treinada não é aproveitada em uma topologia imediatamente maior, durante o processo construtivo. Esta abordagem é explicada em detalhes em [yKY97], e aplicados em trabalhos como [AL08, AL06] e [Yao99].

Com uma abordagem hierárquica o método NNGA-DCOD [AL08] tem como objetivo encontrar redes neurais quase ótimas, otimizando diversos parâmetros livres com o uso de algoritmos genéticos. Recursivamente, o algoritmo busca por pesos iniciais, arquitetura e funções de ativação em diferentes níveis de busca. Na camada de mais alto nível, NNGA-DCOD ainda avalia as aptidões das RNAs obtidas com diversos algoritmos de treinamento. Como característica da maioria dos modelos que exploram diversos parâmetros livres na otimização de RNAs, NNGA-DCOD obteve soluções com excelente capacidade de generalização, embora seja notável o custo computacional que o método demanda.

Por fim, outro trabalho recente e interessante que envolve o ajuste de arquitetura e pesos de RNAs é descrito em [GTT08]. Este trabalho distingui-se dos demais descritos por apresentar duas características peculiares e bem exploradas no contexto de otimização de RNAs: otimiza-

ção multiobjetivo com fronteira de pareto e a decomposição singular de valores para estimação do tamanho da arquitetura da RNA. Este modelo multiobjetivo busca reduzir simultaneamente o erro da rede e a complexidade de sua arquitetura. Para isto, além da soma dos erros quadráticos (SSE) como função de aptidão, foram utilizadas outras duas funções para avaliar a complexidade da rede, combinadas em dois sub-modelos, denominados *HMOEN_HN* e *HMOEN_L2*. O primeiro refere-se à integração da função de aptidão que penaliza a quantidade de neurônios utilizados pela RNA, enquanto o segundo modelo tenta reduzir a complexidade da arquitetura punindo pesos com valores altos. De modo geral, esta metodologia multiobjetivo apresentou resultados bastante relevantes, tanto em capacidade de generalização quanto complexidade de arquitetura, sendo estas características mais notáveis no modelo *HMOEN_L2*.

2.3 Comentários Finais

O uso das Redes neurais artificiais na solução de problemas de inteligência artificial é bastante atrativa. Entretanto, sua aplicação bem sucedida é condicionada a configuração correta de seus parâmetros, funções de ativação e algoritmos de aprendizado. Definir corretamente estas variáveis é uma tarefa difícil, cujas propriedades estarão diretamente relacionadas com o sucesso (ou não) de um projeto. Portanto, é necessário um conhecimento intrínseco sobre o domínio do problema, permitindo a identificação dos principais componentes e desafios das configurações na área de pesquisa de RNAs. A rede MLP, devida a sua ampla aceitação e utilização, foi a topologia escolhida para o desenvolvimento deste trabalho.

A partir dos conceitos fundamentais, enfatizando os maiores desafios na elaboração de um projeto consistente, o conceito de sistemas neurais híbridos é apresentado como alternativa a estas limitações. Diversas pesquisas têm sido apresentadas nos últimos anos na área de pesquisa dos SNHs, geralmente com resultados bons e consistentes. Tendo em vista as vantagens e principalmente limitações das diferentes abordagens, a necessidade de novas pesquisas é evidente. No capítulo seguinte são explicado em detalhes dois métodos evolucionários relativamente recentes, que são utilizados no desenvolvimento desta dissertação. Complementarmente a este capítulo, alguns trabalhos relevantes mais específicos são apresentados.

CAPÍTULO 3

Evolução Diferencial

3.1 Evolução Diferencial

3.1.1 Introdução

A evolução diferencial (DE, acrônimo do nome original em inglês *Differential Evolution*), proposta originalmente por Storn e Price [SP95], é um método baseado em população com busca direta para a otimização de funções contínuas não-lineares e não diferenciáveis. É um algoritmo heurístico de otimização global, trabalha com variáveis de decisão codificadas em números reais e precisa de poucos parâmetros para serem ajustados. DE é simples porém efetivo, superando muitos outros métodos de otimização em termos de velocidade de convergência e robustez, tanto em benchmarks quanto problemas reais [VT04, PSL05].

A respeito de seus operadores de variação, existem diversas modificações do DE propostas tanto pelos autores originais quanto por outros estudos relacionados. Entretanto, o foco desta pesquisa remete ao DE em sua forma clássica e mais popular, conhecido por *DE/rand/1/bin*[1]. O algoritmo é iniciado criando uma população inicial P de tamanho N_p, cujos indivíduos são inicializados aleatoriamente quando nenhuma informação preliminar sobre o espaço de busca está disponível. Para cada indivíduo da população, um novo indivíduo é criado através da soma ponderada da diferença entre dois indivíduos com um terceiro indivíduo, ambos selecionados aleatoriamente e distintos entre si. Este processo se repete enquanto a condição de parada não for alcançada. O pseudocódigo do DE/rand/1/bin é mostrado no alg. 1 e sua descrição geral e respectivos operadores de variação são descritos nas subseções subsequentes.

3.1.2 Mutação

O processo de mutação consiste em escolher aleatoriamente da população três indivíduos e combina-los para gerar um novo indivíduo. Ao contrário de muitos outros algoritmos evolucionários onde a mutação ocorre com baixa probabilidade, no DE este operador de variação

[1] *DE/rand/1/bin* indica que em cada geração os pais são selecionados aleatoriamente, a mutação se da através de 1 subtração e o cruzamento é controlado por experimentações binomiais.

ocorre para cada indivíduo e em todas gerações, sendo assim a mutação uma das operações mais importantes na convergência do processo evolucionário. Além de contribuir aumentando a diversidade da população, a soma ponderada de 1 indivíduo com a diferença entre outros 2 indivíduos tenta gerar um deslocamento da solução candidata em direção ao objetivo, conforme mostra a fig. 3.1. Para cada indivíduo $P_{i,G}$, um novo vetor $V_{i,G}$, é gerado a partir dos indivíduos selecionados pela combinação linear dada por:

$$V_{i,G} = P_{r1,G} + F \times (P_{r2,G} - P_{r3,G}) \tag{3.1}$$

onde $P_{i,G}(i = 1,2,...N_p)$ são os indivíduos da geração G e $r1, r2, r3$ são mutuamente diferentes, selecionados aleatoriamente. F deve estar no intervalo $[0,2]$ e determina a amplificação da diferença entre os dois indivíduos $P_{r2,G} - P_{r3,G}$. Valores maiores de F resultam em uma maior diversidade na população gerada, e valores mais baixos provocam uma convergência mais rápida [SP95, RTS08, Cha08].

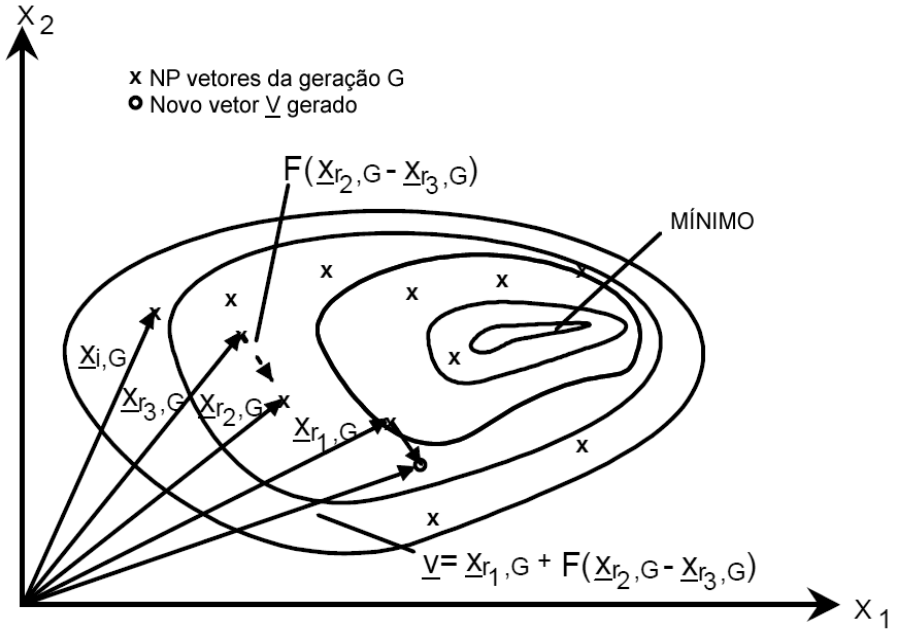

Figura 3.1 Mutação com DE/rand/1/bin.
Função objetivo em um problema bi-dimensional e o processo de geração de um vetor experimental V pelo processo de mutação DE/rand/1/bin. Retirado de [SP95].

3.1.3 Cruzamento

Após a mutação, cada indivíduo $P_{i,G}$ é cruzado com seu respectivo vetor mutante $V_{ri,G}$ gerando uma nova solução candidata $U_{i,G}$, de acordo com a equação abaixo:

$$U_{ji,G} = \begin{cases} V_{ji,G} & if\ rand_j(0,1) < C_r \vee j = k \\ P_{ji,G} \end{cases} \tag{3.2}$$

onde $j = 1,2,...,D$ onde D é a dimensão do problema, ou seja, o tamanho do vetor. Para cada índice do vetor (gene), a nova solução candidata receberá um gene do mutante caso um valor aleatório no intervalo $(0,1)$ for menor que a taxa de cruzamento C_r especificada previamente. De outra forma, $U_{i,G}$ irá receber o gene do indivíduo original. Os valores mais populares para C_r estão no intervalo (0.4, 1).

3.1.4 Seleção

A seleção é o processo que deve decidir qual vetor ($P_{i,G}$ ou $U_{i,G}$) fará parte da próxima geração. Diferentemente de outros algoritmos evolucionários, DE não utiliza hierarquia (elitismo) nem seleção proporcional. Em vez disto, o custo do vetor experimental é calculado e comparado com o custo do vetor alvo . Se o custo do vetor alvo for menor que o custo do vetor experimental, o vetor experimental substitui o vetor alvo na geração seguinte, de acordo com a seguinte regra

$$P_{ji,G} = \begin{cases} U_{ji,G} & if\ f(U_{ji,G}) \leq f(P_{ji,G}) \\ P_{ji,G} \end{cases} . \tag{3.3}$$

Como resultado, esta regra faz com que cada indivíduo seja no mínimo igual ou melhor em relação as gerações anteriores. A ausência de elitismo faz com que o DE seja potencialmente mais rápido que outras abordagens que utilizam este recurso nos operadores de variação. Isso se deve pelo fato que algoritmos de rank aumentam a complexidade do processo demandando mais tempo de processamento. É importante notar que, apesar da ausência de elitismo, o esquema de seleção do DE impede que a melhor solução encontrada seja perdida no processo evolucionário.

Algorithm 1: Pseudo código da evolução diferencial (DE). BFV: Melhor aptidão encontrada; VTR: aptidão objetivo; NFC: contador de épocas; MAX_{NFC}: máximo de épocas previamente definido.

```
1  Inicializar a população P randomicamente a partir de uma distribuição uniforme;
2  enquanto BFV > VTR e NFC < MAX_NFC faça
3      para i = 0 até N_p faça
4          Selecione 3 indivíduos P_i1, P_i2 e P_i3 randomicamente da população atual, onde
           i ≠ i_1 ≠ i_2 ≠ i_3;
5          V_i = P_i1 + F(P_i2 − P_i3);
6          para j = 0 até D faça
7              se rand(0,1) < C_r então
8                  U_{i,j} = V_{i,j};
9              senão
10                 U_{i,j} = P_{i,j};
11             fim
12         fim
13         Analise a aptidão de U_i;
14         se f(U_i) ≤ f(P_i) então
15             P'_i = U_i;
16         senão
17             P'_i − P_i;
18         fim
19     fim
20     P = P';
21     NFC++;
22 fim
```

3.2 Evolução Diferencial Baseada em Oposição

3.2.1 Introdução

Recentemente, Rahnamayan [RTS08] propôs um novo algoritmo denominado Evolução Diferencial Baseada em Oposição (ODE, acrônimo de Opposition-Based Differential Evolution) que integra a meta-heurística aprendizagem baseada em oposição (OBL, acrônimo do nome original em inglês Opposition-Based Learning) [Tiz05] para inicialização e seleção dos indivíduos no DE. Os resultados em simulações baseadas em benchmarks de otimização livre realizadas mostraram que a integração da heurística OBL ao DE acelera a sua convergência, diminuindo a quantidade de chamadas realizadas ao alcançar melhores aproximações nas soluções candidatas. A respeito dos bons resultados, ODE revelou-se uma variação promissora do DE, sendo citado como um dos maiores avanços no campo de pesquisa da evolução diferencial [Cha08].

A aprendizagem baseada em oposição é uma estratégia para o aprendizado de máquina e otimização, pois utiliza os valores, pesos e estados opostos para desenvolver ações no sentido de acelerar as metodologias existentes, tais como redes neurais, aprendizado por reforço e algoritmos baseados em população. A ideia é simples, se não há nenhum conhecimento a priori do espaço de busca do problema, o oposto de uma solução candidata será no pior caso, a própria solução [Tiz05].

Normalmente quando não se tem nenhum conhecimento a priori sobre o problema, os algoritmos baseados em população são inicializados de forma aleatória. Ao realizar aleatoriamente uma busca, é natural que uma solução candidata esteja longe da solução. Em casos extremos, ou seja, se os indivíduos da população estiverem muito longe da solução objetivo, a aproximação demandará muito tempo ou até mesmo ficará intratável [RTS08]. A ideia principal do OBL é poder aumentar a cobertura do espaço de solução, aumentando a precisão e/ou acelerando a velocidade de convergência. Dado um vetor n-dimensional, o seu oposto é definido por:

$$\breve{X}_j = a_j + b_j - X_j \tag{3.4}$$

sendo j um componente do vetor X que esta no intervalo $[a_j, b_j]$. Para a otimização baseada em oposição, levamos em consideração que se a avaliação da qualidade da solução (aptidão) do ponto oposto for melhor que o ponto original, então a solução candidata original é substituída por sua oposta.

De modo geral, uma otimização evolucionária inicia uma população aleatória sem nenhum conhecimento do problema e o processo de busca termina quando algum critério é satisfeito. Utilizando OBL em um otimizador evolucionário, pode-se verificar simultaneamente um indi-

víduo e seu oposto, aumentando as chances de um deles estar perto da solução objetivo. Esta mesma abordagem pode ser aplicada não apenas na inicialização da população, mas também continuamente a cada nova geração do processo evolucionário.

O ODE possui todas as etapas idênticas ao DE, com exceção da inicialização e da geração das novas populações, cujas diferenças são descritas a seguir. Durante a inicialização, os seguintes passos são realizados na inicialização da população baseada em oposição:

1. Inicialize a população $P(N_p)$ randomicamente.
2. Calcule a população oposta de acordo com a fronteiras do espaço de busca pré-estabelecidas.
3. Selecione os N_p melhores indivíduos de acordo com sua aptidão.

Ao longo do processo evolucionário, a cada nova geração a heurística OBL poderá ser novamente utilizada de acordo com uma variável de probabilidade, denominada taxa de *jumping*. Após a seleção, o ODE decorre exatamente como o DE de acordo com seus operadores de variação. Para a criação da população da geração $G+1$, tal como na inicialização, a população oposta é calculada e os N_p melhores indivíduos são escolhidos. A principal diferença entre a seleção e a inicialização esta concentrada no cálculo dinâmico do intervalo para calcular a população oposta. Sendo MIN_j^p e MAX_j^p respectivamente o menor e o maior valor de cada posição j do vetor indivíduo, um ponto oposto é dado por:

$$\breve{P_{i,j}} = MIN_j^p + MAX_j^p - P_{i,j} \tag{3.5}$$

onde $i = 1,2,\ldots,N_p$ e $j = 1,2,\ldots,D$.

Através do calculo dinâmico do intervalo se evitará que a população oposta esteja fora do espaço de busca convergido durante o ciclo de vida do processo evolucionário, o que não ocorreria se o intervalo fosse estático. A taxa de *jumping* é definida no intervalo $jr \in [0,1]$ e determinará a probabilidade que se gerará a população oposta ao longo das gerações. O valor adequado da taxa jumping pode mudar de acordo com o problema, entretanto os autores originais do ODE indicam uma taxa relativamente baixa (0.3) para evitar uma convergência prematura da população[RTS08]. O pseudocódigo do ODE é apresentado no Alg. 2.

3.3 Otimização de Redes Neurais com Evolução Diferencial

Considerando que o treinamento de redes neurais artificiais e a definição de sua arquitetura são problemas de otimização, o DE pode ser aplicado com sucesso a ambos os problemas. No

Algorithm 2: Pseudo código da evolução diferencial baseada em oposição (ODE).

Inicializar a população P randomicamente a partir de uma distribuição uniforme;
`/* Inicialização da população baseada em oposição */`
para $i = 0$ *até* N_p **faça**
  para $j = 0$ *até* D **faça**
    $\breve{P}_{i,j} = a_j^p + b_j^p - P_{i,j}$;
  fim
fim
Selecione os N_p indivíduos mais aptos do conjunto P, $\breve{P}$ como população inicial P;
enquanto $BFV > VTR$ *e* $NFC < MAX_{NFC}$ **faça**
  para $i = 0$ *até* N_p **faça**
    Selecione 3 indivíduos P_{i1}, P_{i2} e P_{i3} randomicamente da população atual, onde $i \neq i_1 \neq i_2 \neq i_3$;
    $V_i = P_{i1} + F(P_{i2} - P_{i3})$;
    para $j = 0$ *até* D **faça**
      se $rand(0,1) < C_r$ **então**
        $U_{i,j} = V_{i,j}$;
      senão
        $U_{i,j} = P_{i,j}$;
      fim
    fim
    Analise a aptidão de U_i;
    se $f(U_i) \leq f(P_i)$ **então**
      $P_i' = U_i$;
    senão
      $P_i' - P_i$;
    fim
  fim
  $P = P'$;
  `/* Geração baseada em oposição */`
  se $rand(0,1) < jr$ **então**
    para $i = 0$ *até* N_p **faça**
      para $j = 0$ *até* D **faça**
        $\breve{P}_{i,j} = MIN_j^p + MAX_j^p - P_{i,j}$;
      fim
    fim
    Selecione os N_p indivíduos mais aptos do conjunto P, $\breve{P}$ como população P;
  fim
  NFC++;
fim

processo supervisionado padrão de treinamento de RNAs, o vetor de entrada X e sua respectiva saída desejada Y são conhecidas. Para treinamento utilizando DE, o objetivo da otimização é ajustar a matriz de pesos W da RNA para diminuir a função de custo, ou seja, o erro de treinamento da saída da rede em relação a Y. Cada indivíduo P_i da população P é um vetor representando uma solução candidata da otimização. No treinamento de RNAs com DE, cada posição do vetor j representa um peso ou bias da rede neural.

Em sua forma mais simples, as configurações dos projetos de treinamento de RNAs com otimizadores evolucionários com codificação em números reais são análogas. Em um primeiro passo, o tamanho da rede neural (quantidade de neurônios ocultos) é pré-definida, geralmente empiricamente. Por ser um algoritmo baseado em população, cada indivíduo representa uma rede neural completa, ou seja, uma possível solução de rede ideal para resolver um determinado problema. Como explicado anteriormente, cada gene, ou seja, cada posição do vetor do indivíduo representa um peso ou bias da RNA. O objetivo do processo evolucionário é minimizar uma função de erro que quantifica a qualidade da rede, através do conjunto de treinamento e validação retirados da base de dados do problema. Para esta função de erro, também chamada de função de custo ou aptidão, geralmente utiliza-se o MSE (Erro Médio Quadrático), SSE (Soma dos Erros Quadráticos) ou CEP (Percentual do Erro de Classificação), de acordo com o objetivo e característica do problema.

O uso do DE como alternativa a métodos convencionais de treinamento de RNAs foi proposto inicialmente por [ML97], indicando sua adequação em termos de velocidade de convergência e qualidade de resposta em redes neurais. Neste trabalho os autores fazem uso de uma versão modificada da evolução diferencial para treinamento de redes neurais de regressão, abrindo novas possibilidades de pesquisa em sistemas neurais híbridos com este algoritmo. Desde então, outras pesquisas buscaram investigar a eficiência do DE como otimizador global de pesos das redes neurais.

O treinamento de RNAs com DE é apresentado de forma similar em diversos outros trabalhos como [IKL03, SB08, Abb02, SA99], revelando um interesse específico da comunidade científica neste assunto. De fato, o treinamento de RNAs é considerado um problema difícil de otimização global, e estudos mais aprofundados que permitam uma maior compreensão das vantagens do uso de algoritmos baseados em população para realizar esta tarefa são frequentemente apresentados. Os resultados apresentados em [IKL03] e [SB08] objetivam comparar a performance do treinamento de MLPs com DE e métodos convencionais baseados em gradiente. Com conclusões semelhantes, estas pesquisas reafirmam a robustez do DE na otimização de pesos, particularmente em redes grandes e em problemas com dados ruidosos ou ausentes.

Com objetivo equivalente, Abdul [AK09] avalia 3 configurações de RNAs para um deter-

minado problema de previsão de séries temporais com configurações distintas de preparação dos dados. Para as simulações, o autor utilizou redes MLP treinadas por DE, MLP e RBF (acrônimo de Radial Basis Function) treinadas por backpropagation. Em todas simulações e cenários, redes MLP treinadas com DE revelaram-se as mais adequadas para este determinado problema, obtendo a melhor precisão.

Como já explicado no capítulo 2, a definição da estrutura de uma RNA para um determinado problema é um passo crítico para o sucesso do projeto. Logo, não é raro encontrar na literatura trabalhos que buscam encontrar uma arquitetura adequada através da otimização evolucionária. Um número relativamente pequeno de trabalhos aplicaram DE com o propósito da otimização da arquitetura de uma RNA, tipicamente reportando bons resultados [Abb02, Abb03, TSHI07, Bhu09]. Takahama [TSHI07] e Bhuiyan [Bhu09] utilizam o DE para aprendizado estrutural de uma RNA, ou seja, dada um arquitetura de tamanho pré-definido, o processo evolucionário tenta eliminar os pesos desnecessários seguindo uma abordagem não invasiva. Este tipo de método baseado na poda da rede, é também explorada com outros algoritmos evolucionários [TS02, TSI02], e possui as desvantagens já citadas no capítulo 2.

Enquanto na bibliografia a maior parte dos trabalhos de otimização estrutural de RNAs com DE focam metodologias de poda e não invasivas em sua forma mais trivial, excepcionalmente o trabalho de Abbass [Abb03] explora a otimização simultânea de pesos e arquitetura de RNAs. Denominado por MPANN (memetic pareto artificial neural network), este trabalho propõe a otimização de RNAs com DE multiobjetivo baseado em fronteira de Pareto para minimizar 2 objetivos de forma simultânea: o erro de classificação e o tamanho da arquitetura da rede. Embora MPANN esteja inserida em uma abordagem não invasiva pois utiliza backpropagation para o treinamento das RNAs, este trabalho possui grande relevância pois considera a dependência entre arquitetura e pesos do processo, evitando a otimização baseada em estágios.

3.4 Comentários Finais

A evolução diferencial [SP95] foi proposta em 1995 e é um dos membros mais novos da família dos algoritmos evolucionários. Apesar de sua simplicidade e robustez, DE foi timidamente explorado nos anos iniciais após sua apresentação. Nos últimos anos, o interesse da comunidade científica por este algoritmo vem aumentando, principalmente a respeito dos muitos casos de sucesso em sua aplicação prática e experimental, demonstrado através da crescente quantidade de publicações em conferências e periódicos sobre o assunto.

A vantagens e limitações da hibridização do DE, principalmente em sistemas conexionis-

tas, é de certa forma uma lacuna a ser preenchida, dadas as raras propostas encontradas na literatura em comparação a demais métodos evolucionários. Propostas relevantes, invasivas e construtivas que concernem o interesse desta dissertação são raras, senão nulas.

Dentre as diversas variações propostas, o ODE [RTS08] destaca-se ao integrar heurística de aprendizado baseado em oposição [Tiz05] para acelerar a convergência do DE, sem alterar as propriedades dos seus operadores de variação e seleção. Com bons resultados apresentados em simulações, ODE foi apresentado no livro de 2008 *Advances in Differential Evolution* [Cha08] como uma das maiores contribuições para o estado da arte na área de pesquisa da evolução diferencial. Não obstante ao respectivo sucesso, a evolução diferencial baseada em oposição é um trabalho recente e não constam estudos que indicam resultados em sistemas neurais híbridos, sendo esta dissertação um primeiro registro de sua aplicação nesta área de pesquisa.

CAPÍTULO 4

Metodologia de Otimização Multimodal de Redes Neurais

4.1 Introdução

O DE é especialmente interessante para a tarefa de treinamento de RNAs pois, ao contrário de outros algoritmos evolucionários, cada indivíduo não verifica apenas um ponto no espaço n-dimensional de forma estocástica. Ao computar a diferença ponderada entre 2 indivíduos, o DE gera uma função gradiente direcionada a solução objetivo [SB08]. Como citado no capítulo 3 , vários estudos investigaram a utilização do DE no treinamento de redes neurais, sendo um dos mais detalhados o de Ilonen *et al* [IKL03]. Neste trabalho, Ilonen *et al* concluem que, apesar dos bons resultados de treinamento com DE, não há evidências significativas de diferença de tempo ou qualidade de solução se comparado com métodos de treinamento por gradiente. Contudo, como o próprio autor conclui, é evidente que a utilização do DE pode ser aplicado com sucesso distinto nestes casos:

- Quando a rede é muito grande;
- Quando frequentemente o problema apresenta mínimos locais;
- Quando há muito ruído nos dados.

Entretanto, a influência dos parâmetros dos operadores de variação F e C_r é pouco explorada no trabalho [IKL03]. De forma geral, os parâmetros F e C_r influenciam na velocidade de convergência e robustez do processo de busca [SP95], sendo o parâmetro de mutação F o principal responsável pelo equilíbrio entre convergência e diversidade. Ao se treinar uma RNA utilizando um algoritmo de aprendizado baseado em gradiente (ex, backpropagation [RHW86]), a taxa de aprendizado precisa ser pequena pois os pesos de uma RNA são sensíveis a mudanças bruscas [BCL07, Hay98]. De forma análoga, a instabilidade na otimização evolucionária dos pesos pode ocorrer quando F assume um valor muito alto. Em contrapartida, quando F é muito pequeno, a diversidade da população cai rapidamente, prejudicando o processo evolucionário.

Na bibliografia pesquisada, os valores mais populares para o parâmetro de mutação estão no intervalo $F \in [0.5, 0.8]$ [SP95, RTS08, VT04, Cha08]. Embora estes valores tenham apresentado melhores resultados em problemas de otimização numérica, neste trabalho verificamos que não são adequados para a tarefa de treinamento de redes neurais. Em testes experimentais, valores de F causaram instabilidade no treinamento, ou seja, na otimização dos pesos da RNA.

Outro problema verificado remete ao tamanho da população. De acordo com os autores originais [SP95] do DE, recomenda-se utilizar uma população de tamanho 5 a 10 vezes a dimensionalidade do problema para uma convergência adequada. A otimização de pesos de RNA's é um problema de alta dimensionalidade, logo seriam necessários muitos indivíduos e consequentemente, excessivo esforço computacional para a tarefa de treinamento.

Para evitar uma convergência prematura utilizando um F pequeno e população reduzida, uma metodologia multimodal pode ser aplicada para explorar diferentes regiões do espaço de busca e também melhorar a diversidade da população. DE, como outros algoritmos baseado em população, pode ser facilmente paralelizado, pelo fato que cada solução candidata da população é avaliada individualmente. Além do mais, esta medida implícita de manutenção da diversidade, também chamada de modelo de ilhas [ES03], tem se tornado cada vez mais atrativa na computação científica de alta performance. Isso se deve por duas principais razões: avanços na tecnologia de multiprocessadores e popularização da computação distribuída [TPPV04].

O presente trabalho utiliza DE paralelo para simultaneamente treinar e selecionar a arquitetura ideal de redes neurais, explorando a dependência entre a arquitetura e a matriz de pesos, e também a propriedade deceptiva das arquiteturas das RNAs. A medida consiste em manter vários grupos (subpopulações) em um único processo evolucionário, restringindo a interação entre indivíduos do mesmo grupo. Em intervalos regulares de tempo (épocas) há uma migração de alguns indivíduos de um grupo para outro. Uma motivação biológica deste modelo de ilhas é conhecida como *Punctuated Equilibria*. Nesta teoria, a estagnação evolucionária de um grupo é interrompida por uma rápida evolução quando a população principal é invadida por indivíduos de outro grupo isolado da mesma espécie [ES03]. Para este processo, é necessário que haja uma separação espacial entre os grupos, porém com algum tipo de comunicação para que alguns indivíduos possam "invadir"outro grupo em intervalos regulares de tempo.

Entretanto, esta pesquisa possui diversas diferenças se comparado a um modelo paralelo convencional. Estas mudanças são propostas perante a necessidade da criação de um modelo construtivo e invasivo para otimização de pesos e arquiteturas de redes neurais. Com base nas discussões anteriores, o modelo multimodal proposto nesta dissertação tem os seguintes objetivos fundamentais:

- Treinar redes neurais artificiais com DE utilizando pequenas populações e parâmetro

de variação F pequeno, obtendo assim um treinamento suave e sem excessivo esforço computacional.

- Manter uma distribuição espacial de populações com arquiteturas distintas com o objetivo de tentar encontrar um conjunto de boas soluções. Esta medida é justificada pela propriedade deceptiva das arquiteturas das RNA's. Assim, o objetivo é encontrar uma arquitetura adequada de baixa complexidade, contornando os problemas de mínimos locais frequentemente encontrados em algorítimos construtivos.
- Permitir o compartilhamento de informação entre as subpopulações. Através da migração periódica, a diversidade de cada subpopulação é aumentada, evitando assim a estagnação evolucionária.
- Sempre que possível, não descartar a informação das buscas anteriores. Como discutido no capítulo 2, várias metodologias construtivas invasivas tendem a descartar a informação de treinamento (otimização de pesos) a medida que a rede cresce. Neste trabalho, as informações de treinamento são compartilhadas e preservadas entre as subpopulações.

4.2 Representação

Neste trabalho, todas RNAs possuem apenas conexões com as camadas adjacentes, apenas uma camada escondida e totalmente conectada. A topologia da rede neural possui N_1 nós de entrada, N_2 nós na camada escondida e N_3 nós de saída. N_1 e N_3 são dependentes do problema, de acordo com o pré-processamento dos dados e do número de atributos de entrada e saída desejada.

Cada solução candidata i é composta por um vetor W de números reais, cada um representando um peso w da rede neural, de acordo com a representação abaixo:

$$W \equiv (W_1, W_2) \tag{4.1}$$

$$W_1 \equiv (w_1, w_2, ..., w_{N_{max1,s}}), w_j \in \Re, j = 1, 2, ..., N_{max1,s} \tag{4.2}$$

$$W_2 \equiv (w_1, w_2, ..., w_{N_{max2,s}}), w_j \in \Re, j = 1, 2, ..., N_{max2,s} \tag{4.3}$$

onde $N_{max1,s} = N_1 \times N_{2,s}$ e $N_{max2,s} = N_{2,s} \times N_3$, e $N_{2,s}$ é o número de neurônios na camada escondida da ilha s.

Cada subpopulação é formada por N_p soluções candidatas, ou seja, vetores de reais que representam as redes neurais, como mostra a figura 4.1. Entretanto, neste trabalho nós utilizamos uma população formada por N_p redes neurais cujas arquiteturas diferem de cada s subpopulação de forma crescente. Ou seja, a subpopulação P_1 é formada por N_p redes neurais com um arquitetura mínima, ex. 1 neurônio na camada escondida, a subpopulação P_2 é formada por N_p redes neurais com 2 neurônios na camada escondida e assim sucessivamente. Dessa forma, o número de neurônios da camada escondida de cada subpopulação é dado por

$$N_{2,s} = \begin{cases} 1 & if\ s = 1 \\ N_{2,s-1} + 1 & \end{cases} . \tag{4.4}$$

Sendo $W_{i,s}$ uma solução candidata da subpopulação P_s, a representação global da população é dada por:

$$P_s \equiv (W_{1,s}, W_{2,s}, ..., W_{N_p,s}), W_{i,s} \in \Re, i = 1, 2, ..., N_p \tag{4.5}$$

$$Pop \equiv (P_1, P_2, ..., P_k), P_s \in \Re, s = 1, 2, ..., k \tag{4.6}$$

onde k é a quantidade de subpopulações definida.

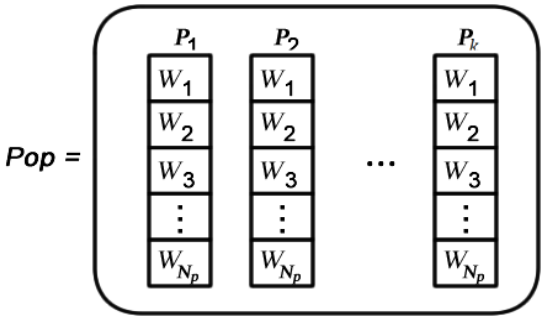

Figura 4.1 Representação da população.

4.3 Migração

De forma similar a todos os algoritmos paralelos baseados em população, cada subpopulação converge para uma solução de forma independente e apenas compartilhando a informação da busca em períodos regulares. Este processo, denominado migração no modelo de ilhas, consiste na troca de indivíduos entre subpopulações. Periodicamente, alguns indivíduos (usualmente

2-5) de uma subpopulação migram para outra, aumentando a diversidade e possivelmente melhorando a convergência [ES03]. Entretanto, ao se utilizar subpopulações com indivíduos de tamanhos diferentes, este processo não pode ser realizado diretamente.

Para resolver este problema, é proposta a utilização de migração unidirecional, trocando indivíduos da subpopulação com menores indivíduos para a subpopulação subseqüente com indivíduos imediatamente maiores (com 1 neurônio a mais), como mostra a Figura 4.2. Para isto, os indivíduos que irão migrar recebem 1 neurônio inativo em sua arquitetura, ou seja, novas conexões com valor zero (Figura 4.3).

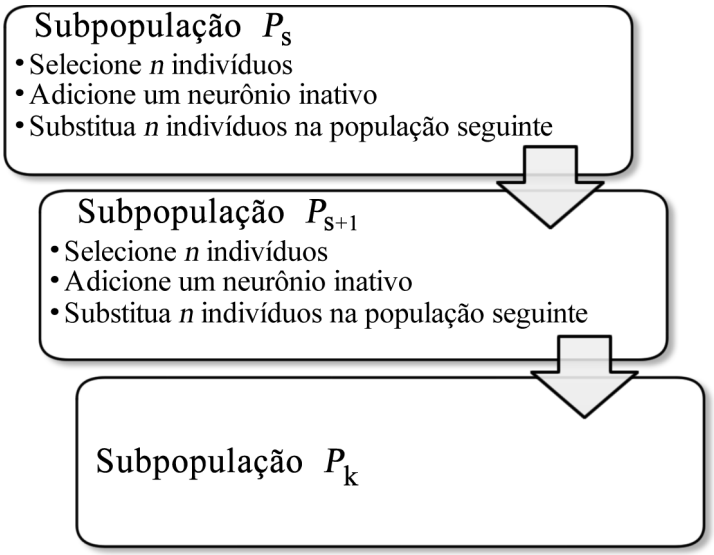

Figura 4.2 Migração unidirecional de $S \in 1, 2, ..., k$ subpopulações.

Em cada subpopulação, *n* indivíduos são selecionados, um neurônio é adicionado com conexões sinápticas setadas em zero e então migram para o grupo imediatamente maior em representação. É fácil perceber que a última subpopulação (que possui a maior representação de arquitetura) não migrará indivíduos, e que a primeira subpopulação (com indivíduos com menor arquitetura) será a única que não receberá indivíduos de outras subpopulações.

Também é importante destacar que as conexões do neurônio inativo não são fixas. Isso quer dizer que essas novas redes que migrarão participarão normalmente do processo evolucionário, sofrendo mutação e cruzamento com os demais indivíduos. A opção de se adicionar um neurônio nulo justifica-se pelo objetivo de se preservar a capacidade de generalização da arquitetura original. Com este neurônio nulo adicional a rede crescerá sem alterar sua performance, permitindo assim, integrar-se normalmente ao processo evolucionário com redes imediatamente maiores. Na prática, estes indivíduos procedentes da migração influenciarão tanto na função

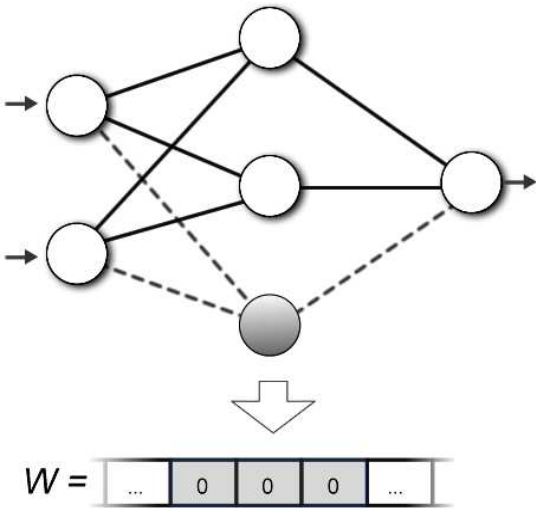

Figura 4.3 Representação de um indivíduo da população.
Adição de um neurônio com conexões nulas.

de diversificar a subpopulação, quanto na opção de nova direção evolucionária no espaço de busca.

4.4 Crescimento da Rede

O processo evolucionário conseguirá manter a diversidade da população trocando indivíduos entre as subpopulações em intervalos de épocas pré-definidos e simultaneamente conseguirá mensurar qual a subpopulação que consegue o melhor resultado após o treinamento, obtendo assim um conjunto de boas soluções. Desta forma, uma estimativa do tamanho da arquitetura necessária para a convergência da rede é dada de acordo com o melhor indivíduo da melhor subpopulação. Entretanto, esta abordagem pré-definida limita a busca de arquiteturas no intervalo $[1, k]$, o que pode ser insuficiente para resolver um determinado problema.

Após a convergência do processo evolucionário de acordo com o critério de parada, sabe-se qual o melhor indivíduo da população e o melhor indivíduo de cada grupo. Sob o contexto desta metodologia, se o melhor indivíduo de P_1 for o pior em comparação aos melhores indivíduos das outras subpopulações, é provável que a arquitetura mínima contida em P_1 não seja suficiente para generalizar bem o problema. Desta forma, P_1 é excluída e uma nova subpopulação P_{k+1} é criada, com um neurônio a mais que a representação contida em P_k, como mostra o Algoritmo 3. Sendo assim, P_1 agora terá 2 neurônios na camada oculta, P_2 terá 3 e assim sucessivamente.

Este processo é repetido até que o melhor indivíduo de P_1 não seja o pior se comparado aos melhores indivíduos das demais subpopulações.

4.5 Critério de Parada

A evolução diferencial, assim como demais otimizadores baseados em população, necessita de um critério pré definido que informe ao processo evolucionário quando a busca deverá ser interrompida. Tradicionalmente, os critérios mais utilizados em otimização numérica são um número máximo de funções de aptidão chamadas, número máximo de épocas ou ainda tempo de execução. Ao se treinar redes neurais com algoritmos evolucionários, uma medida oportuna é utilizar a resposta da rede frente ao conjunto de treino e validação para mensurar quando a otimização esta estagnada e deve ser encerrada.

O critério de parada do treinamento utilizado neste trabalho é baseado no critério proposto por [PHU05]. Durante o processo evolucionário, a aptidão de cada indivíduo é dada por alguma medida de erro dos dados de treinamento (ex: MSE, SSE, NMSE, etc.). Sendo cada indivíduo uma rede neural, apresentam-se os dados de treinamento para obter o erro correspondente à sua aptidão.

Para evitar excesso de treinamento, a cada 10 gerações apresenta-se ao melhor indivíduo da população o conjunto de validação e verifica-se a presença de *overfitting* através de:

$$overfitting = 100 * \left(\frac{fittestVal_i + 1}{bestVal_j + 1} - 1 \right) \tag{4.7}$$

onde $bestVal_j$ é o melhor resultado de validação encontrado, usando

$$bestVal_j = Min(\forall(fittestVal_i)), i = 1, 2, \ldots, j. \tag{4.8}$$

Se a rede não estiver generalizando mais, um contador é incrementado, senão este mesmo contador é zerado, como mostra o Algoritmo 4. Quando o contador chegar ao valor 30, o treinamento para. Este processo é baseado na observação que um bom resultado de treinamento não implica necessariamente em uma boa generalização, de acordo com os resultados obtidos no conjunto de validação.

Com este critério de parada, a melhor rede será encontrada 300 gerações antes de terminar o treinamento, mas desta forma é possível obter uma solução final com boa capacidade de

Algorithm 3: pseudo-código otimização multimodal

Entrada: $k \leftarrow$ número de subpopulaçoes, com $k > 1$;
$n \leftarrow$ número inicial de neurônios na camada escondida;
$N_p \leftarrow$ número de indivíduos em cada subpopulação;
$m \leftarrow$ passo de migração;
Resultado: *bestNet*

$n' \leftarrow n$;
$n \leftarrow n+1$;
repita
  $n' \leftarrow n$;
  $n \leftarrow n+1$;
  para $s = 1$ *até* k **faça**
    se P_{s+1} *não existe* **então**
      inicializa a população P_s com N_p indivíduos com n' neurônios;
    senão
      $P_s \leftarrow P_{s+1}$;
    fim
    $n' \leftarrow n'+1$;
  fim
  enquanto *critério de parada não for encontrado* **faça**
    Realize uma época do DE em cada subpopulação;
    Avalie todos indivíduos de todas subpopulações;
    se *epoch é múltiplo de m* **então**
      Realize a migração unidirecional;
    fim
  fim
  $worst \leftarrow 1$;
  $best' \leftarrow$ aptidão do melhor indivíduo da população P_1;
  $bestNet \leftarrow$ melhor indivíduo da população P_1;
  para $s = 1$ *até* k **faça**
    $best_s \leftarrow$ aptidão do melhor indivíduo da população P_s;
    se $best_s > best'$ **então**
      $worst \leftarrow s$;
      $best' \leftarrow best_s$;
      $bestNet \leftarrow$ melhor indivíduo da população P_s;
    fim
  fim
até $worst \neq 1$;

generalização.

Algorithm 4: Critério de parada utilizado para interromper a otimização de cada subpopulação

Entrada: fittestVal atual, bestVal, maxOverFitCount, overFitCount
Resultado: novo bestVal
Saída: "VERDADEIRO"ou "FALSO"

```
1  início
2      overfitting ← 100 * ((fittestVal+1)/(bestVal+1) − 1);
3      se overfitting ≥ 0 então
           /* Há overfittnes ou não há mudanças na aptidão.
              Incrementa o contador                          */
4          overFitCount++;
5          se overFitCount ≥ maxOverFitCount então
               /* Para o treinamento                         */
6              retorna VERDADEIRO;
7          fim
8      senão
           /* Encontrou uma solução melhor.  Atualiza bestVal e
              zera overFitCount                              */
9          bestVal ← fittestVal;
10         overFitCount ← 0;
11         retorna FALSO;
12     fim
13 fim
```

4.6 Funções de Custo

Durante a evolução, a função de custo (aptidão) é o que determina a sobrevivência ou não de um indivíduo ao longo das gerações. Do ponto de vista do problema, a aptidão representa a qualidade da solução apresentada por cada indivíduo, indicando o andamento da busca. No escopo dos sistemas neurais híbridos, o objetivo do processo evolucionário é minimizar a função de custo, ou seja, reduzir o erro da RNA de acordo com o conjunto de dados apresentado.

A medida de performance da RNA utilizada nesta dissertação é descrita pela equação:

$$E(T) = CEP + MSE. \tag{4.9}$$

onde T é o conjunto de treinamento, MSE é o erro médio quadrático e CEP é o erro percentual de classificação, definido por:

$$CEP = \frac{100}{\#T} \sum_{x_{\in T}} \varepsilon(x). \tag{4.10}$$

onde $\#T$ é o número de exemplares do conjunto T e $\varepsilon(x)$ é o exemplo classificado corretamente.

Adicionalmente, nós combinamos a medida de erro com 2 técnicas conhecidas de regularização rede, como tentativa de melhorar a capacidade de generalização. Um termo β é adicionado a função de custo original $E(T)_0$:

$$E(T) = E(T)_0 + \mu\beta. \tag{4.11}$$

onde μ é o parâmetro que define a importância do termo de regularização β, podendo ser:

Decaimento de Pesos (WD, do inglês *Weight Decay*) tenta diminuir uniformemente todos os pesos da rede neural adicionando um termo de punição no erro global da rede, conforme a equação 4.12. O decaimento de pesos tem como objetivo diminuir o ruido e a variância na saída da rede possivelmente causados por pesos com valores absolutos altos que podem ocorrer durante o treinamento. A regularização WD é dada por:

$$\beta = \frac{\|w\|^2}{\psi}. \tag{4.12}$$

onde w é o vetor de pesos da RNA. Assumindo que estamos avaliando simultaneamente redes com diferentes arquiteturas, nós dividimos o termo de punição pelo número de conexões (pesos) ψ da arquitetura, evitando assim erros maiores em arquiteturas maiores.

A outra função de regularização investigada, *Eliminação de Pesos* (WE, do inglês *Weight Elimination*), utiliza uma função de custo para tentar reduzir o número de conexões. Como mostra a equação 4.13, pesos menores são reduzidos mais rapidamente que pesos maiores [ZL07]. De acordo com [Hay98], estes pesos pequenos tendem a resultar em uma generalização ruim, dada a probabilidade de eles assumirem valores totalmente arbitrários. A regularização WE é dada por:

$$\beta = \sum_{i \in_{total}} \frac{(w_i/w_0)^2}{1 + (w_i/w_0)^2}. \tag{4.13}$$

onde w_0 é um parâmetro pré-definido.

4.7 Comentários Finais

O modelo paralelo apresentado, objetivo primordial desta dissertação, propõe uma metodologia invasiva e construtiva para treinamento e seleção de arquitetura de redes neurais. Distintamente, o algoritmo busca contornar alguns dos problemas referenciados em sistemas neurais híbridos e beneficiar-se das vantagens naturais do paralelismo na otimização evolucionária. Como descrito nas seções anteriores, o paralelismo aplicado refere-se a cada população evoluir de forma independente, ou seja, de granularidade grossa. O uso da técnica multimodal implícita em um sistema neural híbrido, a forma de busca por arquiteturas e pesos da RNA e o esquema de migração unilateral são algumas das características que diferenciam este trabalho dos demais trabalhos encontrados na literatura.

É importante perceber que a metodologia permite configurações distintas, de acordo com o problema a ser resolvido. Na descrição da representação (seção 4.2), o tamanho inicial da rede na primeira subpopulação é apresentado com apenas 1 neurônio na camada escondida. Entretanto, nada impede que esta arquitetura mínima inicial seja maior, caso o projetista tenha algum conhecimento a priori do problema e conclua que a rede precisará de muitos neurônios para obter uma solução plausível. A atribuição miníma da arquitetura de forma adequada evitará crescimento desnecessário da rede (seção 4.4), exigindo menos tempo para a convergência.

Da mesma forma, as seções 4.3 e 4.4 demostram como a metodologia proposta utiliza o paralelismo para explorar a característica deceptiva das arquiteturas da RNA. Em suas respectivas descrições, as arquiteturas das subpopulações diferem em apenas 1 neurônio oculto com as populações subsequentes. Embora o intervalo de exploração de arquiteturas é dependente do número de ilhas, o projetista pode optar em manter uma diferença estrutural maior entre cada subpopulação, como múltiplos de 2 ou 3 neurônios a mais de forma crescente.

Finalmente, duas versões do modelo multimodal foram implementadas, uma utilizando evolução diferencial e a outra evolução diferencial baseada em oposição. Para fins investigativos, três funções de custo diferentes são aplicadas de forma independente, totalizando seis configurações distintas de aplicação do modelo, como detalha a tabela 4.1.

Método Evolucionário	**Função de Custo**	**Normalização de Pesos ($\mu\beta$)**
DE	$E(T) = CEP + MSE$	-
DE	$E(T) = E(T)_0 + \mu\beta$	Decaimento de Pesos
DE	$E(T) = E(T)_0 + \mu\beta$	Eliminação de Pesos
ODE	$E(T) = CEP + MSE$	-
ODE	$E(T) = CEP + MSE + \mu\beta$	Decaimento de Pesos
ODE	$E(T) = CEP + MSE + \mu\beta$	Eliminação de Pesos

Tabela 4.1 Configurações utilizadas na metodologia multimodal proposta.

CAPÍTULO 5

Experimentos

5.1 Introdução

Após a apresentação do modelo multimodal de otimização simultânea de pesos e arquiteturas, uma série de experimentos se faz necessária para análise de sua performance, vantagens e limitações. Com este objetivo, este capítulo tem a finalidade de descrever os experimentos realizados, objetivos buscados e bases de dados utilizadas. Além de uma avaliação quantitativa, o propósito dos experimentos também atende a necessidade de comparar os resultados obtidos com outras técnicas encontradas na literatura.

5.2 Domínios de Problemas

Para o desenvolvimento dos experimentos, 4 problemas bem conhecidos de classificação foram utilizados, listados na tabela 5.1. As bases de dados foram retiradas dos repositórios UCI [AN07] e PROBEN1 [Pre94]. A escolha destes problemas para a validação do modelo proposto se deve à popularidade de estudos com os mesmos domínios, permitindo uma posterior avaliação comparativa. Outro critério que motiva a escolha destas bases é suas distintas características de tamanho, complexidade, número de classes e atributos.

5.3 Configuração Geral dos Experimentos

Existem diversas estratégias recomendadas para análise de performance de um método. Algumas das sugestões são listadas no relatório PROBEN1 [Pre94] e utilizadas na realização dos experimentos deste trabalho. Para o treinamento e posterior análise de performance, cada base de dados é dividida em um conjunto de treinamento, validação e teste, onde cada conjunto possui 50%, 25% e 25% dos exemplos respectivamente. Em cada experimento, os dados foram particionados randomicamente com estratificação, ou seja, mantendo a proporção de exemplos por classe. Excepcionalmente para o problema Câncer, foram utilizadas as 3 permutações su-

Base de Dados	**Composição**			**Descrição**
	Dados	Atributos	Classes	
Câncer	699	9	2	Refere-se ao diagnóstico do câncer de mama, distribuído em 2 classes, benigno (65,5% dos dados) e maligno (34,5%).
Diabetes	768	8	2	Baseado em resultados de exames médicos, tenta classificar se um indivíduo está com diabetes ou não. Distribuição das classes: 65.1% dos exemplos são negativo para diabetes e 34.9% para positivo.
Tireoide	7200	21	3	Contém informações para descobrir disfunções da tireoide. Distribuição das classes: a classe hipertireoide representa 2.3% dos exemplos, a classe hipotireoide representa 5.1% e tireoide normal representa 92.6% dos casos.
Iris	150	4	3	Os atributos dos dados objetivam classificar 3 tipos da planta Íris. Possui 50 instâncias de cada flor, ou seja, distribuição de $1/3$ por classe.

Tabela 5.1 Descrição dos domínio de problemas utilizados nos experimentos, quantidade de atributos e informações adicionais.

geridas no relatório PROBEN1 [Pre94], com o objetivo de se realizar comparações similares com os resultados contidos neste relatório e em outros trabalhos que utilizam esta mesma metodologia. Os experimentos com as 3 permutações da base câncer foram realizadas 30 vezes cada. Nas demais bases de dados, utilizou-se validação cruzada 30×10 *folds*, estratificadas e randomicamente particionadas.

Como já relatado, o modelo proposto foi concebido para otimizar redes do tipo MLP. Na literatura diversas pesquisas fazem uso de redes MLP com este mesmo domínio de problema, permitindo uma análise qualitativa de performance. A topologia da MLP utilizada nos experimentos é totalmente conectada, possui apenas uma camada oculta e todos nodos computáveis possuem como função de ativação a função sigmoide logística:

$$f(net) = \frac{1}{1+e^{-net}}. \tag{5.1}$$

Para os experimentos com o modelo construtivo proposto, o arquitetura inicial da rede MLP possui apenas 1 neurônio na camada oculta. Os parâmetros utilizados foram escolhidos arbitrariamente, mas com o propósito de verificar a performance do método com uma população pequena e taxa de mutação menor que a recomendada na literatura, conforme discutido no capítulo 4. Os parâmetros em todos experimento foram definidos como segue:

- Número de subpopulações: 3
- Número de indivíduos em cada subpopulação: 30
- Passo de migração: 50 épocas
- Número de indivíduos que migram: 5 melhores
- F (Fator de mutação): 0.3
- C_r (Taxa de crossover): 0.7
- J_r (jumping rate - ODE): 0.3

Para comparar e verificar a relevância estatística entre a performance dos métodos, foram realizados testes de hipótese *t-test* emparelhado. Este teste é realizado a partir dos resultados apresentados nas múltiplas execuções de cada base de dados. O objetivo é verificar a diferença estatística entre as seis configurações propostas para a metodologia multimodal. O intervalo de confiança utilizado foi de 95% bicaudal.

A implementação do modelo para avaliação experimental foi realizada através da ferramenta Matlab. Foram desenvolvidos *scripts* para todas as partes do modelo híbrido, dispensando o uso de qualquer *toolbox* que integra a ferramenta. Os experimentos foram executados em um computador com processador Dual-Core 2.8GHZ com 4GB de memória RAM, na plataforma *Windows*.

5.4 Influência da taxa de mutação no treinamento de RNAs

Conforme discutido nos capítulos anteriores, o processo de mutação na evolução diferencial é um dos mais importantes operadores de variação do algoritmo. Sua configuração é dada pelo ajuste do parâmetro livre F, que determina a ponderação da diferença entre 2 indivíduos ao

somar com um terceiro indivíduo. Enquanto estudos indicam um valor sugerido de F compreendido no intervalo $[0.5, 0.8]$ para otimização numérica [SP95, RTS08, VT04, Cha08], as conexões sinápticas de uma RNA são muito sensíveis a mudanças bruscas, o que pode ocorrer quando este parâmetro de amplificação é muito alto.

A taxa de mutação F utilizada no DE e ODE, explicada na seção 3.1.2, é diretamente responsável pela velocidade de convergência e diversidade da população. De acordo com estudos voltados à otimização numérica, se o parâmetro é muito baixo a convergência é mais rápida, mas a diversidade da população cai abruptamente, prejudicando a qualidade da resposta [SP95, RTS08]. Dessa forma, este experimento tem como objetivo fazer uma análise preliminar da influência deste parâmetro no treinamento de RNAs, verificando a qualidade de resposta utilizando diversos valores para este parâmetro. Apesar do método multimodal proposto ter como um dos objetivos aumentar a diversidade da população e dessa forma permitir uma taxa de mutação F menor, é necessário um estudo inicial para buscar um *trade-off* adequado, principalmente pelo fato de que não existem estudos anteriores que investigam sua influência no treinamento de redes neurais.

Dados	Parâmetro F	Erro (Treino)	Tempo (s)
Câncer 1	0.1	3.19	**7.96**
	0.3	**1.82**	8.43
	0.5	2.98	8.04
	0.8	3.2	9.24
Câncer 2	0.1	2.11	**7.04**
	0.3	1.76	7.53
	0.5	**1.72**	8.06
	0.8	1.98	7.2
Câncer 3	0.1	1.88	11.3
	0.3	**1.44**	10.27
	0.5	1.5	**6.98**
	0.8	1.67	7.66

Tabela 5.2 Influência da taxa de mutação no treinamento de RNAs nas 3 permutações do problema do câncer.

Para a realização deste experimento preliminar foram utilizadas as três permutações da base

de dados Câncer [Pre94] para treinamento de uma RNA com arquitetura fixa através da evolução diferencial. Cada experimento foi executado dez vezes e os resultados correspondem às médias das execuções. Durante o treinamento, foram analisadas a curva de erro de treinamento e também o tempo de convergência, utilizando o mesmo critério de parada da metodologia multimodal, explicada na seção 4.5 do capítulo 4. A rede utilizada possui 4 neurônios na camada oculta e foram analisadas quatro configurações diferentes do parâmetro de mutação F do DE: $0.1, 0.3, 0.5, 0.8,$.

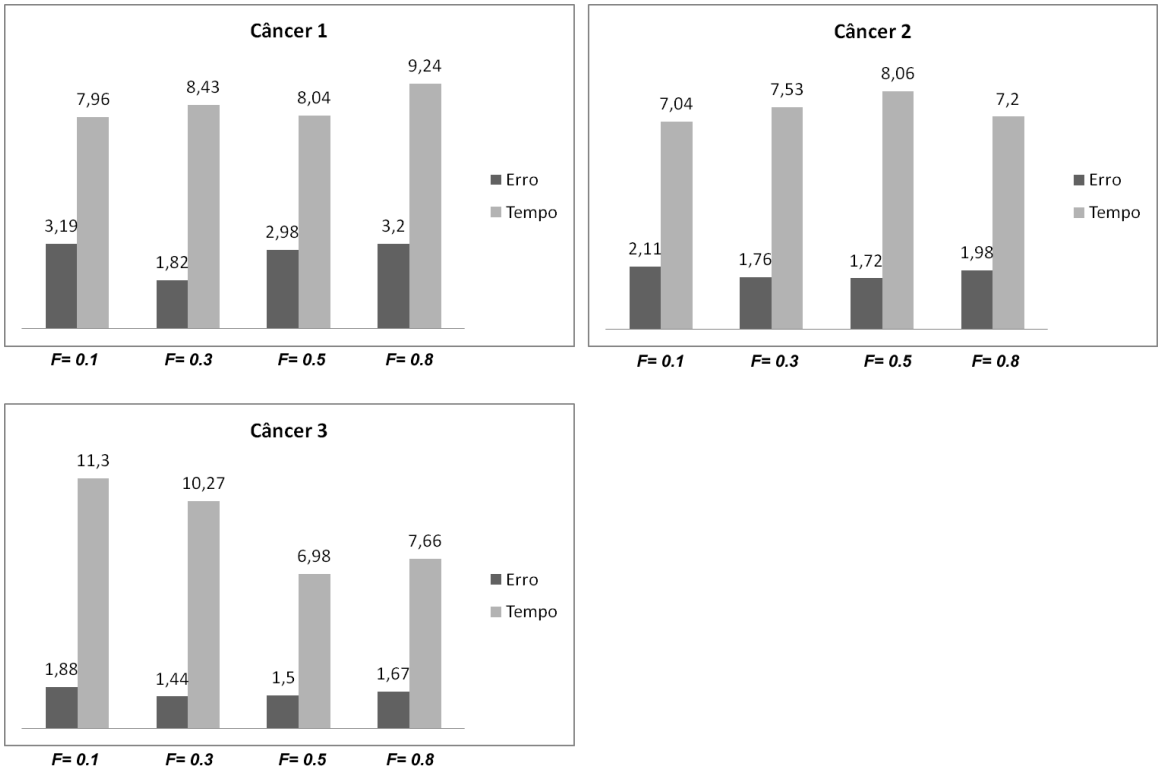

Figura 5.1 Influência da taxa de mutação no treinamento de RNAs.

Apesar deste experimento não ter a intenção de ser exaustivo ao utilizar apenas um domínio de problema, ele serve como referência empírica para o ajuste da taxa de mutação nos demais experimentos. A tabela 5.2 mostra os resultados obtidos, de erro de treinamento e tempo de convergência. Pode-se perceber que a configuração do parâmetro $F = 0.3$ obteve menor erro de treinamento em 2 das 3 simulações (permutações 1 e 3). A configuração $F = 0.5$ obteve bom desempenho na permutação 2, ligeiramente melhor que $F = 0.3$. Em relação ao tempo, o taxa de mutação $F = 0.1$ convergiu mais rapidamente que as demais configurações em 2 das 3 permutações, reforçando a hipótese sobre velocidade de convergência com taxas baixas

explicadas anteriormente. Uma visualização melhor dos resultados obtidos é dada no gráfico comparativo 5.1.

Apesar deste experimento não ser conclusivo, é fácil perceber que valores de F com bom desempenho em problemas de otimização numérica não indicam necessariamente um boa configuração para se treinar redes neurais. Como exemplo, um dos valores muito utilizados na literatura ($F = 0.8$) [SP95, RTS08], obteve um dos piores desempenhos, tanto em erro de treinamento quanto velocidade de convergência. Logo, reafirma-se a hipótese considerada sobre a sensibilidade dos pesos da RNA em mudanças bruscas, alterando significativamente a qualidade da resposta quando a taxa F é muito alta. Para analisar esta suposição, o comportamento da curva de erro de treinamento de uma execução nas 300 primeiras gerações de cada configuração é mostrada na figura 5.2.

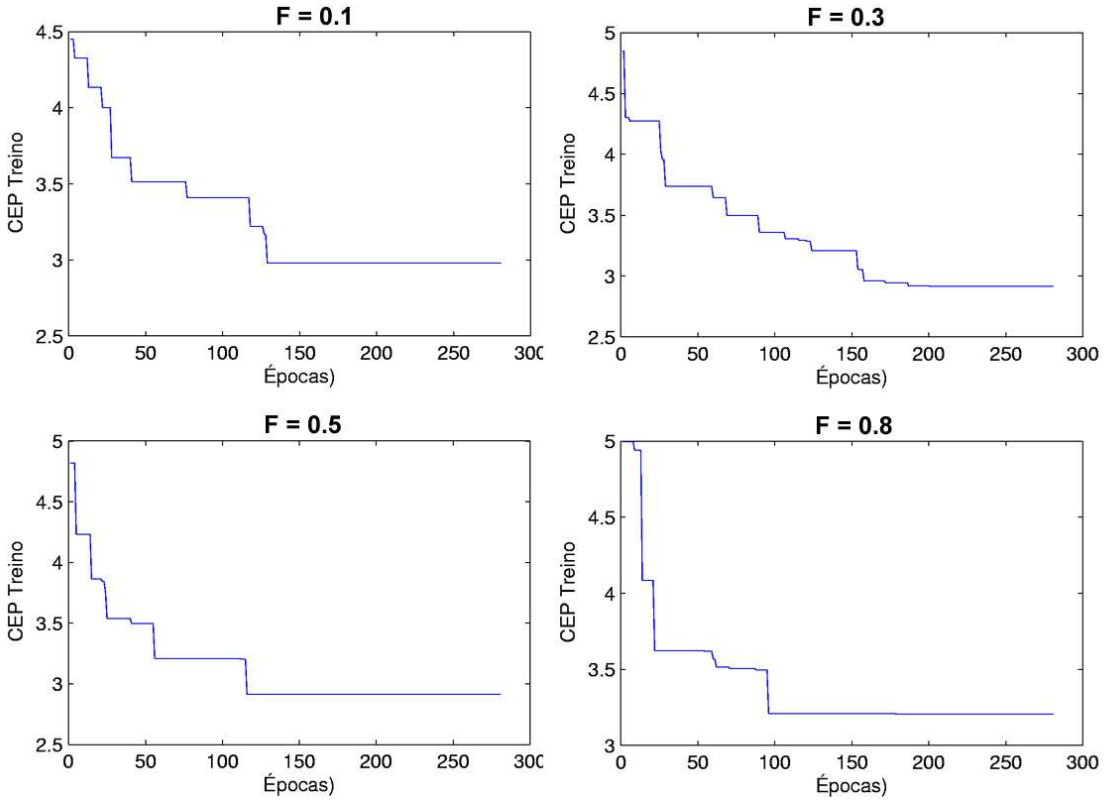

Figura 5.2 Evolução do treinamento em diferentes taxas de F.

Pelo comportamento do gráfico nas diferentes taxas de mutação, algumas informações podem ser extraídas. Em taxas altas ($F \in [0.5, 0.8]$), o processo evolucionário diminui rapidamente o erro de treinamento nas primeiras gerações, seguido de longos períodos de estagnação.

Nestes valores altos, o erro de treino varia em grandes saltos, enquanto nos valores de F reduzidos, a curva de treinamento é mais suave. Pode-se perceber pela análise do gráfico que a taxa $F = 0.1$ reduz o erro de modo razoavelmente constante, entretanto, o treinamento é visivelmente estagnado antes da geração 150, o que não ocorre no ajuste $F = 0.3$, que treina uniformemente por mais gerações. Com base nesta série de análises, conclui-se que o treinamento de RNAs não precisa (e não deve) ter uma taxa de mutação alta. Por outro lado, valores mínimos também restringem o processo evolucionário. A partir deste pressuposto e em concordância com as características buscadas no modelo proposto, a taxa de mutação escolhida para a execução dos experimentos foi $F = 0.3$.

Como explicado anteriormente, este experimento serve apenas como apoio preliminar para guiar a configuração da taxa de mutação da evolução diferencial nos demais experimentos de performance. De qualquer forma, pelo fato de haver uma carência de estudos anteriores desta importante configuração para realizar o treinamento de RNAs, este experimento torna-se significativamente relevante ao contribuir com o propósito geral do objetivo desta dissertação: otimização de RNAs com evolução diferencial.

5.5 Capacidade de Generalização e Complexidade da Arquitetura

Ao se propor sistemas neurais híbridos, este deve estar justificado de acordo com a performance de suas características, onde é primordialmente considerada a qualidade de resposta. Esta seção descreve o conjunto de experimentos que tem maior relevância com o propósito desta dissertação: capacidade de generalização e tamanho da arquitetura das soluções encontradas pelo modelo proposto.

Nesta série de experimentos, foram utilizados todos os conjuntos de dados e configurações especificadas na seção 5.3. O objetivo deste conjunto de experimentos é analisar a capacidade de generalização, arquitetura encontrada (quantidade de neurônios ocultos) e tempo de busca da metodologia multimodal de otimização de pesos e arquiteturas proposta. Para ambas as técnicas, os experimentos foram executados com e sem uso de normalização dos pesos. Portanto, esta série também permite uma avaliação sobre a influência do decaimento de pesos e eliminação de pesos nesta metodologia construtiva proposta.

5.5.1 Resultados para o problema do Câncer

Para este experimento foram utilizadas as três permutações propostas no relatório PROBEN1 [Pre94] com o objetivo de realizar uma posterior comparação com outros trabalhos, incluindo os reportados no referido relatório. Portanto, os resultados de performance e testes estatísticos foram agrupados e apresentados por permutação, denominadas *câncer1*, *câncer2* e *câncer3*. É importante ressaltar que para todos os experimentos, os resultados apresentados correspondem às médias das execuções. Para a descrição dos erros médios de treinamento, validação e teste, é utilizada a medida de custo *CEP* que corresponde ao erro de classificação percentual. Para fins de simplificação nas tabelas de resultados, as metodologias multimodais com DE e ODE foram denominadas pelos acrônimos MDE e MODE, respectivamente.

A tabela 5.3 apresenta os resultados obtidos através da metodologia multimodal proposta com a permutação *câncer1*, utilizando DE, ODE e também com o uso de regularização dos pesos. Analisando os resultados, percebe-se que o uso da regularização WD (*decaimento de pesos*) beneficia significamento a capacidade de generalização neste problema, tanto para o MDE quanto para o MODE. É interessante notar que a melhor qualidade de resposta em relação às demais configurações não se deve pela exploração de uma região melhor do espaço de buscas, pois o erro de treinamento apresentado com decaimento de pesos não é menor do que nas outras simulações. Em contrapartida, a regularização por *eliminação de pesos* (WE) prejudicou a convergência da RNA, obtendo erros de treinamento elevados principalmente no DE. Utilizando ODE, a regularização WE conseguiu uma sensível melhora de qualidade de resposta no conjunto de teste, em relação à metodologia sem regularização.

Algoritmo	Erro				Arq.	Tempo (seg.)
	Treino	Validação	Teste (CEP)	σ		
MDE	2.95	1.91	1.56	0.69	3.30	36.89
MDE:WD	3.05	1.84	**0.98**	0.51	**3.16**	48.64
MDE:WE	3.62	2.20	1.85	0.66	3.23	**30.57**
MODE	2.89	1.94	1.70	0.53	**3.06**	**67.94**
MODE:WD	3.04	1.86	**1.57**	0.78	3.33	155.56
MODE:WE	3.43	2.01	1.64	0.75	3.50	96.98

Tabela 5.3 Resultados da metodologia proposta para o problema Câncer 1. CEP: erro de classificação médio; σ: desvio padrão da classificação; Arq: número de neurônios na camada oculta; Tempo(seg): tempo de convergência em segundos.

O melhor resultado no conjunto de teste em todas as simulações foi obtida pela metodologia aplicada com MDE e regularização WD, com 0.98% de erro de classificação. Entretanto, o bom

desempenho foi obtido sob o custo do pior tempo médio de execução em relação ao não uso de regularização e regularização com WE. Outra análise importante refere-se ao tempo de convergência das três configurações do MODE. Como explicado no capítulo 3, o algoritmo ODE foi proposto como uma variação do DE, tendo como um dos seus objetivos, acelerar a convergência em relação ao algoritmo original. Contudo, o uso do ODE na metodologia proposta (MODE) não só obteve resultados piores como também demandou mais tempo de processamento, comparando com o tempo médio apresentados nas três configurações da metodologia com DE (MDE).

Os resultados encontrados em todas configurações da metodologia apresentaram arquiteturas simples e similares, com poucos nodos na camada intermediária. Pode-se concluir com este experimento que a metodologia foi consistente na busca de arquiteturas, mas com significativa discrepância de erro obtido e tempo de processamento entre as configurações propostas. Esta variação de resultados se deve na maior parte pelas características inerentes do DE e ODE na otimização dos pesos, além das influências das técnicas de regularização.

O objetivo da tabela 5.4 é expor a análise estatística que verifica a diferença de resultados das configurações propostas nos experimentos com a base *câncer1*. Os valores expressos nesta tabela correspondem ao *P-value* do *t-test*, ou seja, a probabilidade da hipótese nula ser verdadeira. Essa verificação foi realizada com as informações do erro de teste e busca avaliar se a superioridade de um modelo possui significância estatística. Vale lembrar que a hipótese nula afirma que os resultados médios de todas configurações são iguais, e que sua veracidade é determinada pela probabilidade *P-value*, variando no intervalo ente 0 e 1.

Alg.	MDE	MDE:WD	MDE:WE	MODE	MODE:WD	MODE:WE
MDE	-	0.0007 ↑	0.1046 ↕	0.3869 ↕	1 ↕	0.7046 ↕
MDE:WD	-	-	0.0001 ↓	0.0001 ↓	0.0025 ↓	0.0005 ↓
MDE:WE	-	-	-	0.3333 ↕	0.1618 ↕	0.3173 ↕
MODE	-	-	-	-	0.4154 ↕	0.7590 ↕
MODE:WD	-	-	-	-	-	0.7193 ↕
MODE:WE	-	-	-	-	-	-

Tabela 5.4 Analise estatística dos resultados do problema do câncer permutação 1. *P-value* da hipótese nula do *teste-t*

A tabela 5.4 confirma o bom resultado obtido por MDE:WD. Isso quer dizer que o uso da regularização WD na metodologia MDE é estatisticamente superior aos resultados apresentados pelas outras configurações. Na configuração MODE:WD, o erro foi obtido foi menor que MODE e MODE:WE, entretanto não há evidência estatística significativa que possa afirmar sua

superioridade. Por fim, é interessante notar que estatisticamente os erros obtidos com MDE e MODE:WD são idênticos, de acordo com o *t-test*.

Os resultados obtidos com a permutação *câncer2* são apresentados na tabela 5.5. Distintamente dos erros de teste da base anterior, o uso da regularização *decaimento de pesos* teve pouca contribuição para a melhoria da qualidade das respostas. Os melhores resultados foram obtidos por MDE e MDE:WD, com 4.5% de erro de classificação no conjunto de teste. Da mesma forma, resultados muito semelhantes foram apresentados por MODE e MODE:WD, indicando que esta regularização teve pouca ou nenhuma contribuição para metodologia nesta base de dados. Por outro lado, a regularização *eliminação de pesos* influenciou negativamente a qualidade da resposta com ODE, obtendo erros maiores.

Algoritmo	Erro				Arq.	Tempo (seg.)
	Treino	Validação	Teste (CEP)	σ		
MDE	1.78	1.49	**4.50**	0.64	3.26	**37.12**
MDE:WD	1.93	1.47	4.50	0.66	**3.00**	48.38
MDE:WE	2.17	1.59	5.18	0.98	3.26	39.05
MODE	1.86	1.48	4.65	0.83	3.63	57.52
MODE:WD	1.89	1.47	**4.64**	0.70	3.2	**53.93**
MODE:WE	2.01	1.53	5.2	1.15	**3.16**	62.39

Tabela 5.5 Resultados da metodologia proposta para o problema Câncer 2. CEP: erro de classificação médio; σ: desvio padrão da classificação; Arq: número de neurônios na camada oculta; Tempo(seg): tempo de convergência em segundos.

Para validar a performance de generalização de cada configuração, a tabela 5.6 apresenta os resultados do teste de hipótese dos erros de teste da permutação *câncer2*. Como pode ser observado, as configurações que apresentaram os melhores resultados,MDE e MDE:WD, são idênticas estatisticamente. Similarmente, não há distinção estatística entre a qualidade das respostas obtidas por MODE e MODE:WD. Como dito anteriormente, a metodologia com uso de da regularização WE não conseguiu bons resultados, hipótese reafirmada pela análise comparativa da tabela 5.6.

Assim como na permutação *câncer1*, as arquiteturas encontradas para resolver a permutação *câncer2* nas diversas configurações foram bastante semelhantes, indicando que metodologia multimodal precisa de poucos neurônios na camada oculta para a convergência do problema. A configuração MDE:WD obteve o menor erro médio e também a menor arquitetura entre todas simulações, com arquitetura média de 3 neurônios.

Para encerrar a análise de performance da metodologia multimodal proposta com o problema do Câncer, a tabela 5.7 apresenta os resultados obtidos com permutação *câncer3*. Nova-

Alg.	MDE	MDE:WD	MDE:WE	MODE	MODE:WD	MODE:WE
MDE	-	1 ↕	0.0034 ↓	0.3676 ↕	0.4997 ↕	0.0144 ↓
MDE:WD	-	-	0.0022 ↓	0.4888 ↕	0.4866 ↕	0.0061 ↓
MDE:WE	-	-	-	0.0423 ↑	0.0204 ↑	0.9499 ↕
MODE	-	-	-	-	0.9363 ↕	0.0460 ↓
MODE:WD	-	-	-	-	-	0.0226 ↓
MODE:WE	-	-	-	-	-	-

Tabela 5.6 Analise estatística dos resultados do problema do câncer permutação 2. *P-value* da hipótese nula do *teste-t*

mente, o *decaimento de pesos* contribuiu para a capacidade de generalização do metodologia com DE. Obtendo o melhor desempenho de classificação, MDE:WD apresentou um erro de 3.32% no conjunto de teste, o menor erro entre todas simulações. Entretanto, o benefício do WD não é replicado na metodologia quando utilizado ODE, onde o maior erro de teste foi obtido por MODE:WD, com um erro médio de 3.86%.

Algoritmo	Erro				Arq.	Tempo (seg.)
	Treino	Validação	Teste (CEP)	σ		
MDE	1.48	2.95	3.48	0.76	3.53	33.85
MDE:WD	1.45	3.03	**3.32**	1.01	3.33	47.83
MDE:WE	2.12	3.19	3.76	0.81	**3.14**	**26.70**
MODE	1.46	2.98	3.73	0.98	3.2	48.77
MODE:WD	1.73	3.18	3.86	0.93	3.1	74.23
MODE:WE	2.09	3.15	**3.42**	0.92	**3.06**	**45.15**

Tabela 5.7 Resultados da metodologia proposta para o problema Câncer 3. CEP: erro de classificação médio; σ: desvio padrão da classificação; Arq: número de neurônios na camada oculta; Tempo(seg): tempo de convergência em segundos.

Outra novidade refere-se à regularização WE nos resultados do *câncer3*, que apresentou respostas relativamente boas. Nas 3 configurações da metodologia que utilizam ODE, MODE:WE apresentou o menor erro de classificação. A significância estatística destes resultados reportados são analisados na tabela 5.8. Como pode-se perceber, a metodologia MDE:WD obteve melhor resultado que todas as demais configurações, com exceção de MODE:WE, cujas respostas no conjunto de teste apresentam grande semelhança estatística.

Os resultados apresentados para as 3 permutações do problema do Câncer tiveram como propósito avaliar a performance geral da metodologia multimodal proposta em diversas configurações. Após os erros serem apresentados e discutidos, é importante que se faça um compara-

Alg.	MDE	MDE:WD	MDE:WE	MODE	MODE:WD	MODE:WE
MDE	-	0.2821 ↕	0.1110 ↕	0.3733 ↕	0.1101 ↕	0.8242 ↕
MDE:WD	-	-	0.1115 ↕	0.1236 ↕	0.0399 ↓	0.6151 ↕
MDE:WE	-	-	-	0.8727 ↕	0.7023 ↕	0.1332 ↕
MODE	-	-	-	-	0.5616 ↕	0.2084 ↕
MODE:WD	-	-	-	-	-	0.0406 ↓
MODE:WE	-	-	-	-	-	-

Tabela 5.8 Analise estatística dos resultados do problema do câncer permutação 3. *P-value* da hipótese nula do *teste-t*

tivo com outros trabalhos encontrados na literatura. Com este propósito, a tabela 5.9 apresenta os melhores resultados médios obtidos pela metodologia proposta utilizando DE e ODE em comparação com outros resultados reportados. Para manter uma coerência de comparação, estes trabalhos foram escolhidos por utilizarem as mesmas permutações do problema do câncer. As informações a respeito do tempo necessário para a convergência foram omitidas pois estas medidas não foram reportadas nos trabalhos comparados.

		L-BP	P-BP	N-BP	MGNN [PHU05]	MDE	MODE
Cancer 1	Teste (CEP)	2.93	1.47	1.38	3.14	0.98	1.57
	Teste σ	0.18	0.60	0.49	0.01	0.51	0.78
	Arq.	0	4+2	4+2	10	3.16	3.33
Cancer 2	Teste (CEP)	5.00	4.52	4.77	6.41	4.50	4.64
	Teste σ	0.61	0.70	0.94	0.01	0.64	0.70
	Arq.	0	8+4	8+4	10	3.26	3.20
Cancer 3	Teste (CEP)	5.17	3.37	3.70	4.61	3.32	3.42
	Teste σ	0.00	0.71	0.52	0.01	1.01	0.92
	Arq.	0	4+4	4+4	10	3.33	3.06

Tabela 5.9 Comparação de desempenho com outros métodos encontrados na literatura para o problema do Câncer.

O dados apontados na tabela 5.9 referem-se a alguns dos melhores resultados apontados no relatório PROBEN1 [Pre94], onde o autor buscou manualmente diversas configurações topológicas de RNAs. O trabalho MGNN[PHU05] refere-se a uma método invasivo de otimização de redes neurais, conforme explica a seção 2.2.2.

De modo geral, a metodologia multimodal proposta apresentou um bom resultado, com boa capacidade de generalização e arquiteturas pequenas. Apesar de não haver uma análise

comparativa de tempo de execução com outros métodos, os tempos apresentados pela metodologia são bastante aceitáveis, precisando geralmente de poucos segundos para obter uma boa performance neste problema. Com base nos resultados apresentados, é importante notar que a regularização *decaimento de pesos* contribuiu significativamente para a qualidade das respostas enquanto utilizado DE. De fato, os melhores resultados obtidos nas 3 permutações foram com a configuração MDE:WD. Outro ponto que merece atenção é o baixo desempenho obtido pela metodologia utilizando ODE. Em nenhum dos casos foi possível obter uma resposta superior com este algoritmo, tanto em capacidade de resposta quanto velocidade de convergência.

5.5.2 Resultados para o problema Diabetes

A tabela 5.10 apresenta os resultados gerais obtidos com a metodologia multimodal proposta em suas diferentes variações. O problema diabetes é expressivamente mais complexo que o do câncer, como pode-se averiguar nos erros de classificação e em trabalhos correlatos exemplificados na tabela 5.12. Os melhores desempenhos obtidos neste domínio de problema com a metodologia foram conseguidas através do DE sem regularização e com regularização da rede (WD). As configurações MDE e MDE:WD obtiveram o mesmo erro de classificação no conjunto de teste (24.38%), com arquiteturas igualmente semelhantes.

Algoritmo	Erro				Arq.	Tempo (seg.)
	Treino	Validação	Teste (CEP)	σ		
MDE	19.56	20.56	**24.38**	1.81	**3.19**	17.01
MDE:WD	19.62	21.31	**24.38**	2.16	3.34	**16.32**
MDE:WE	20.80	20.45	24.74	2.22	3.33	16.96
MODE	19.53	19.09	24.56	2.47	3.36	20.73
MODE:WD	19.75	20.85	**24.50**	2.61	3.18	**20.39**
MODE:WE	19.66	20.82	24.59	2.53	**3.1**	20.89

Tabela 5.10 Resultados da metodologia proposta para o problema Diabetes. CEP: erro de classificação médio; σ: desvio padrão da classificação; Arq: número de neurônios na camada oculta; Tempo(seg): tempo de convergência em segundos.

O uso do ODE novamente não apresentou resultados melhores, em relação ao uso do DE. Entretanto, neste problema os erros de classificação obtidos com o conjunto de teste para todas configurações foram muito semelhantes, como pode-se comprovar pela análise do *t-test* demostrado na tabela 5.11. A partir destas informações, pode-se perceber que a única diferença estatisticamente significante está nos resultados apresentados por MDE e MDE:WE. Tanto neste quanto no domínio de problema anterior, as configurações que fizeram uso da regularização

eliminação de pesos tiveram performance geral inferior em relação as demais. Estes resultados indicam que o WE não provê contribuições para o modelo construtivo e otimização dos pesos, enquanto WD obtém resultados no mínimo iguais aos melhores resultados de todas configurações.

Alg.	MDE	MDE:WD	MDE:WE	MODE	MODE:WD	MODE:WE
MDE	-	0.9777 $\updownarrow$	0.0052 $\downarrow$	0.2759 $\updownarrow$	0.5333 $\updownarrow$	0.3365 $\updownarrow$
MDE:WD	-	-	0.0097 $\downarrow$	0.1340 $\updownarrow$	0.1431 $\updownarrow$	0.0913 $\updownarrow$
MDE:WE	-	-	-	0.4048 $\updownarrow$	0.3427 $\updownarrow$	0.4063 $\updownarrow$
MODE	-	-	-	-	0.8178 $\updownarrow$	0.9097 $\updownarrow$
MODE:WD	-	-	-	-	-	0.7211 $\updownarrow$
MODE:WE	-	-	-	-	-	-

Tabela 5.11 Analise estatística dos resultados do problema Diabetes. *P-value* da hipótese nula do *teste-t*

A metodologia proposta obteve resultados satisfatórios com o problema do Diabetes, como pode-se averiguar comparando com alguns resultados reportados na literatura e apresentados na tabela 5.12. Um dos resultados mais favoráveis refere-se ao tempo de convergência. Para obter o melhor resultado na configuração MDE, foram necessários 17 segundos de otimização. No entanto, a análise comparativa desta e de outras medidas de performance com os resultados dos trabalhos relacionados deve ser cautelosa. Isto porque as estratégias de particionamento dos dados e execução dos experimentos diferem entre uma técnica e outra. De qualquer forma, o objetivo da tabela 5.12 é servir como um referencial, apresentando resultados alcançados em trabalhos recentes.

Algoritmo	Erro		Arq.	Tempo
	Teste (CEP)	σ		
MDE	24.38	1.81	3.19	17 seg.
MODE:	24.50	2.61	3.18	20 seg.
GaTSa [ZL07]	27.06	-	1.20	-
CNNDA [IM01]	19.91	2.08	4.3	14 seg.
OMWTandARC [LYZ06]	25.87	-	4.53	-
NNGA-DCOD [AL08]	15.92	-	5.20	30.90 horas

Tabela 5.12 Comparação de desempenho com outros métodos encontrados na literatura para o problema do Diabetes.

Em relação à arquitetura, o método GaTSa [ZL07] obteve a menor rede, mas com erro expressivamente maior que os demais trabalhos. As arquiteturas encontradas com a metodologia proposta são ligeiramente menores que a obtida na metodologia OMWTandARC [LYZ06],

assim como o erro de classificação do conjunto de teste. O melhor resultado de classificação é reportado no método NNGA-DCOD [AL08]. Sua superioridade é justificada pela estrutura hierárquica de otimização de RNAs, ajustando diversos parâmetros livres adicionalmente à arquitetura e conjunto de pesos. Entretanto, os bons resultados são alcançados sob o custo de uma lenta convergência, precisando de cerca de 30 horas para a otimização.

5.5.3 Resultados para o problema Íris

O problema Íris é considerado simples em relação ao conjunto de dados pois os exemplos são distribuídos na mesma proporção entre as 3 classes do problema. No entanto, são fornecidas apenas 150 amostras dos dados, dificultando o aprendizado e o problema como um todo. Os resultados obtidos pela metodologia proposta é apresentado na tabela 5.13, para as 6 configurações especificadas. Como pode-se avaliar preliminarmente, o erro no conjunto de teste é relativamente baixo de forma geral, mas o desvio padrão apresentado é expressivamente alto, superando até mesmo o erro de classificação. Esta variação nas respostas é dependente da partição estratificada randomicamente, indicando uma desarmonia entre os atributos comuns dos exemplos da base de dados.

Como mostra a tabela de resultados 5.13, a metodologia proposta utilizando ODE aparentemente obteve uma melhor performance de classificação no conjunto de testes. Mas como o desvio padrão foi muito alto nas simulações, esta aparente superioridade não é ratificada pela análise estatística dos resultados pelo *t-test*, conforme indica a tabela 5.14. De fato, todos resultados apresentados pelas 6 configurações são potencialmente semelhantes em termo de capacidade de generalização

Algoritmo	Erro				Arq.	Tempo (seg.)
	Treino	Validação	Teste (CEP)	σ		
MDE	0.66	1.39	3.35	3.14	3.82	**15.12**
MDE:WD	1.00	1.22	**3.20**	3.11	**3.69**	19.88
MDE:WE	1.02	1.56	3.33	3.18	**3.69**	16.85
MODE	0.61	1.49	3.26	3.19	4.00	**20.87**
MODE:WD	0.79	1.15	3.34	3.21	**3.79**	24.85
MODE:WE	0.77	1.59	**3.15**	2.89	3.99	25.39

Tabela 5.13 Resultados da metodologia proposta para o problema Iris. CEP: erro de classificação médio; σ: desvio padrão da classificação; Arq: número de neurônios na camada oculta; Tempo(seg): tempo de convergência em segundos.

Assim como nos problemas anteriores, a metodologia proposta obteve uma arquitetura mé-

dia simples e semelhante em todas simulações, sendo esta uma das características mais marcante do modelo proposto. Em relação ao tempo, as simulações convergiram em tempo aceitável com pouca distinção entre as configurações com e sem regularização da rede. Contudo, novamente as configurações MODE exigiram mais tempo que os modelos MDE, reforçando a hipótese de que a heurística para acelerar a convergência não compensa a complexidade do algoritmo.

Alg.	MDE	MDE:WD	MDE:WE	MODE	MODE:WD	MODE:WE
MDE	-	0.2682 ↕	0.9051 ↕	0.6618 ↕	0.8114 ↕	0.6618 ↕
MDE:WD	-	-	0.2708 ↕	0.4433 ↕	0.5987 ↕	0.5911 ↕
MDE:WE	-	-	-	0.4751 ↕	0.7453 ↕	0.9391 ↕
MODE	-	-	-	-	0.7199 ↕	0.7024 ↕
MODE:WD	-	-	-	-	-	0.4450 ↕
MODE:WE	-	-	-	-	-	-

Tabela 5.14 Analise estatística dos resultados do problema Iris. *P-value* da hipótese nula do *teste-t*

A tabela 5.15 objetiva referenciar alguns resultados expostos em trabalhos recentes que utilizam o problema Íris. Apesar de estatisticamente semelhantes, foram escolhidas uma configuração de MDE e outra MODE, escolhidas arbitrariamente pelo critério de erro de classificação no conjunto de teste. Embora não tenha conseguido equiparar os resultados com o melhor trabalho HMOEN_L2 [GTT08], a metodologia proposta apresentou resultados relativamente bons, tanto em capacidade de generalização quanto complexidade da arquitetura. Infelizmente, os trabalhos referenciados não reportaram os tempo de execução, para fins comparativo. Ainda que estes trabalhos possam utilizar diferentes estratégias de configuração de seus experimentos, a tabela 5.15 é um guia interessante para validar os resultados obtidos no problema Íris como relevantes.

Algoritmo	Erro		Arq.	Tempo (seg.)
	Teste (CEP)	σ		
MDE	3.20	3.11	3.69	19 seg.
MODE:	3.15	2.89	3.99	25.39
GaTSa [ZL07]	5.25	-	4.13	-
HMOEN_HN [GTT08]	9.00	0.03	2.68	-
HMOEN_L2 [GTT08]	2.00	0.02	3.08	-
OMWTandARC [LYZ06]	4.61	-	2.65	-

Tabela 5.15 Comparação de desempenho com outros métodos encontrados na literatura para o problema Íris.

5.5.4 Resultados para o problema Tireoide

O problema da tireoide caracteriza-se como o mais difícil problema de classificação utilizado para validar esta proposta. A conjunto de dados da tireoide é abundante, com 7200 exemplos disponíveis. Entretanto, a distribuição deste exemplos nas 3 classes do problema é altamente desproporcional. De acordo com o relatório PROBEN1, um bom classificador deve obter uma precisão de classificação significantemente melhor que 92% [Pre94].

Dada a complexidade do problema, o modelo multimodal proposto obteve uma ótima performance de classificação neste problema, conforme apresenta a tabela 5.16. O melhor resultado de treinamento, validação e teste foi obtido pela configuração MDE com 2.32%, 2.79% e 3.04% de erro de classificação, respectivamente. Pelo fato da base de dados ser altamente desbalanceada, as técnicas de regularização da rede WE e WD não apresentaram bons resultados, tanto utilizando DE quanto ODE. O bom desempenho do MDE é comprovado pela pela análise do *t-test* dada na tabela 5.17. Através dos valores de probabilidade da hipótese-nula, percebe-se que a superioridade do MDE em relação às demais configurações possui alta significância estatística.

Algoritmo	Erro				Arq.	Tempo (seg.)
	Treino	Validação	Teste (CEP)	σ		
MDE	2.32	2.79	3.04	1.60	5.48	787.67
MDE:WD	7.13	7.59	7.12	0.45	3.88	259.96
MDE:WE	7.45	7.68	7.38	0.31	3.85	234.08
MODE	3.72	4.44	4.33	1.73	5.46	715.39
MODE:WD	7.37	7.93	7.40	0.29	3.95	247.50
MODE:WE	7.94	7.37	7.35	0.30	3.06	256.46

Tabela 5.16 Resultados da metodologia proposta para o problema Tireoide. CEP: erro de classificação médio; σ: desvio padrão da classificação; Arq: número de neurônios na camada oculta; Tempo(seg): tempo de convergência em segundos.

A configuração MODE apresentou um erro de classificação razoável (4.33%), mas ainda assim significativamente inferior que o obtido por MDE. Em relação à arquitetura, as duas configurações que apresentaram as melhores respostas também apresentaram as maiores redes. No entanto, estas redes mais complexas indicam que a metodologia de busca de arquiteturas de forma construtiva foi desempenhada com sucesso nas configurações ODE e MODE, dada a dificuldade do problema. As redes mais simples encontradas pelas metodologias com regularização não apresentaram complexidade suficiente para o problema, fato verificado pelo erro de treinamento. Dessa forma, é possível colocar como evidência que, para este problema, a

regularização da rede não somente interferiu na eficácia da busca por pesos como também prejudicou o sistema híbrido como um todo, interrompendo a busca correlacionada de arquiteturas e pesos precocemente.

Alg.	MDE	MDE:WD	MDE:WE	MODE	MODE:WD	MODE:WE
MDE	-	0.0001↓	0.0001 ↓	0.0023 ↓	0.0001 ↓	0.0001 ↓
MDE:WD	-	-	0.1603 ↕	0.0001 ↑	0.1979 ↕	0.0831 ↕
MDE:WE	-	-	-	0.0001 ↑	0.1603 ↕	0.5712 ↕
MODE	-	-	-	-	0.0001 ↓	0.0001 ↓
MODE:WD	-	-	-	-	-	0.0831 ↕
MODE:WE	-	-	-		-	-

Tabela 5.17 Analise estatística dos resultados do problema da Tireoide. *P-value* da hipótese nula do *teste-t*

O tempo necessário para a convergência do problema da tireoide é muito maior em comparação aos problemas anteriores. No entanto, é correto afirmar que este aumento se deve em grande parte pelo tamanho da base de dados e também pela alta dimensionalidade do conjunto de atributos. Ainda é importante destacar que os tempo obtidos por MDE:WD, MDE:WE, MODE:WD e MODE:WE são significativamente menores, mas não por uma possível hipótese de melhor desempenho algorítmico. Ao analisar os resultados, percebe-se que estes tempos inferiores foram ocasionados pelo interrompimento erroneamente antecipado do processo evolucionário.

Algoritmo	Erro		Arq.	Tempo (seg.)
	Teste (CEP)	σ		
MDE	3.04	1.60	5.48	787 seg
MODE:	4.33	1.73	5.46	715 seg
GaTSa [ZL07]	7.15	-	7.12	-
PNN [Tem09]	5.19	-	-	-
OMWTandARC [LYZ06]	7.33	-	6.39	-

Tabela 5.18 Comparação de desempenho com outros métodos encontrados na literatura para o problema da Tireoide.

Após a apresentação dos resultados apontados com a metodologia proposta, a tabela 5.18 introduz uma comparação de alguns trabalhos que buscaram resolver o problema da tireoide, citados e descritos na seção 2.2.2. Adicionalmente, foram comparados os resultados com o melhor desempenho obtido no trabalho [Tem09], cujo enfoque esta concentrado no uso de redes MLP e probabilísticas para o problema da tireoide. Dentre os trabalhos citados, o método multi-

modal proposto obteve uma qualidade de resposta nitidamente melhor, embora estes resultados reportados não permitam uma análise estatística conclusiva. No entanto esta comparação é importante para exemplificar e ratificar o bom resultado obtido pelo método proposto em relação a estes trabalhos. A configuração MDE da metodologia proposta obteve não somente o menor erro de classificação do conjunto de teste, como o fez com a menor arquitetura, comparando com os dados disponíveis desta referência.

5.6 Comparação com Busca Manual de Arquiteturas

Além da busca multimodal construtiva de arquiteturas, o modelo proposto utiliza paralelismo e migração para melhorar a diversidade da população e possivelmente explorar outras regiões do espaço de busca. Para verificar a influência destes processos na performance da rede, foram realizados experimentos de otimização de pesos de RNAs com Evolução Diferencial utilizando busca manual de arquiteturas. Nesta série de experimentos, redes neurais foram treinadas com DE, utilizando os mesmos parâmetros e problemas utilizados pela metodologia proposta, porém com arquiteturas fixas. Na busca manual de arquiteturas, os experimentos foram executados 30 vezes em redes com $2, 4, 6, 8$ e 10 neurônios na camada oculta. Os valores apresentados correspondem as médias do erro de classificação da melhor arquitetura encontrada manualmente para cada problema.

A tabela 5.19 apresenta os melhores resultados obtidos no treinamento com DE e busca manual de arquiteturas comparados com os resultados obtidos pela metodologia proposta, em todos problemas. Nas três permutações do problema do Câncer, os erros de classificação do conjunto de teste na busca manual foram ligeiramente maiores em relação ao modelo multimodal. Como pode ser observado entretanto, existe uma discrepância significativa no tamanho da arquitetura encontrada. Na permutação *Câncer 1*, a arquitetura com busca manual que obteve o melhor resultado possui 8 nodos na camada oculta para um erro de classificação de 1.70%. A metodologia proposta obteve um erro médio de 1.56% com uma arquitetura média de 3.3 nodos. Na permutação *Câncer 2*, os erros de classificação foram semelhantes na busca manual e na metodologia multimodal, com arquiteturas de 4 nodos e 3.26 nodos respectivamente. A permutação *Câncer 3* também apresentou erros de classificação semelhantes, porém com uma diferença significativa de arquiteturas encontradas, com 3.56 nodos na metodologia proposta e 6 nodos através da busca manual.

No problema da Diabetes, a melhor arquitetura encontrada pela busca manual foi de 8 nodos. A metodologia multimodal encontrou uma arquitetura média de 3.19 nodos, mesmo

assim com erros de classificação ligeiramente melhores e menor desvio padrão. No problema da Íris, a busca manual conseguiu resultados um pouco melhores, com uma arquitetura de 10 nodos. Contudo, a metodologia proposta encontrou arquiteturas muito menos complexas, com uma média de 3.82 nodos. Para o problema da Tireoide, a melhor arquitetura por busca manual foi de 6 nodos, similar com a média encontrada pelo metodologia, de 5.48 nodos. Neste problema, mesmo com arquitetura um pouco menor, a metodologia proposta obteve resultados significativamente melhores, com erros de classificação no conjunto de teste de 3.04% contra 6.21% de erro através da busca manual.

Resultados		MDE	DE
Cancer 1	Teste (CEP)	**1.56**	1.70
	Teste σ	0.69	0.62
	Arquitetura	**3.3**	8
Cancer 2	Teste (CEP)	**4.5**	4.51
	Teste σ	0.64	0.81
	Arquitetura	**3.26**	4
Cancer 3	Teste (CEP)	**3.48**	3.52
	Teste σ	0.76	0.75
	Arquitetura	**3.53**	6
Diabetes	Teste (CEP)	**24.38**	24.53
	Teste σ	1.81	2.53
	Arquitetura	**3.19**	8
Íris	Teste (CEP)	3.35	**3.25**
	Teste σ	3.14	3.00
	Arquitetura	**3.82**	10
Tireoide	Teste (CEP)	**3.04**	6.21
	Teste σ	1.60	0.86
	Arquitetura	**5.48**	6

Tabela 5.19 Comparação de desempenho com RNAs treinadas por DE e busca manual de arquiteturas.

5.7 Comentários Finais

Os problemas utilizados nos experimentos possuem características distintas entre si, como quantidade de atributos, número de classes e quantidade de exemplos, permitindo dessa forma, obter uma visão ampla da performance da metodologia proposta. Adicionalmente, foi possível analisar e comparar a eficiência das duas bases evolucionárias utilizadas na metodologia, além da influência das técnicas de regularização. Para agregar confiabilidade nas comparações, utilizou-se validação cruzada e *t-test* nos experimentos realizados.

Com base nos resultados, a primeira constatação relevante é a deficiência da variante Evolução Diferencial baseada em Oposição em relação ao algoritmo evolucionário original. Em todos experimentos realizados, o ODE não conseguiu propor uma qualidade de resposta melhor que o DE. O ODE apenas igualou a acurácia de classificação do DE no problema da Íris e foi sensivelmente pior em todos os demais. Considerando que o ODE foi proposto como uma variação do DE para acelerar sua convergência, esta expectativa foi igualmente frustrada neste trabalho. De modo geral, a metodologia com ODE precisou de mais tempo para convergência em todos cenários, mostrando que complexidade adicional do algoritmo reduz o número de épocas [RTS08], mas exige mais tempo computacional para obter uma solução. No entanto, esta complexidade pode ser diluída no processo evolucionário e apresentar vantagem em bases grande e complexas, como ocorreu no problema da Tireoide, único caso em que o ODE foi mais rápido.

Como dito anteriormente, este tempo adicional de convergência do ODE é hipoteticamente causado pelas heurísticas incluídas no algoritmo original da evolução diferencial. Como pode ser observado no algoritmo 2, que descreve o modelo ODE, o algoritmo eventualmente duplica toda sua população para cálculo da população oposta. Com esta estratégia, o ODE necessita de memória adicional para alocação destes novos indivíduos, além de computar repetidamente, ao longo das gerações, as fronteiras do espaço de busca através dos maiores e menores valores de cada atributo de toda a população. Outro fator que contribui negativamente para a velocidade de convergência do ODE é a necessidade de ordenar pela aptidão os indivíduos da população original e da oposta, acrescentando ao modelo a complexidade de algoritmos de *ranking*. O fato do ODE ter obtido resultados piores pode ser justificado pela perda de generalização de forma acentuada no processo evolucionário, pois suas heurísticas buscam justamente reduzir a exploração do espaço de busca, tentando assim, diminuir o número de épocas necessárias para convergência. No entanto, observou-se empiricamente que em problemas de otimização evolucionária de dimensionalidade alta, a manutenção da diversidade influencia diretamente a qualidade da resposta obtida.

A regularização Eliminação de pesos (WE) é geralmente utilizada para podar redes neurais, eliminando conexões desnecessárias. A influência desta regularização no método construtivo proposto foi investigada nos experimentos e mostrou-se pouco eficaz na tarefa de beneficiar a generalização da rede. Por outro lado, a regularização Decaimento de Pesos (WD) contribuiu para aumentar a precisão da rede em boa parte dos experimentos, principalmente no problema do Câncer. Ambas regularizações foram ineficazes no problema da Tireoide, com alto erro de classificação tanto no treinamento quanto no conjunto de teste. A partir destas evidências, é possível perceber que utilizar estas técnicas de regularização visando uma maior capacidade de generalização depende de um conhecimento preliminar do problema. Com base nestes resultados, percebe-se que WD é pouco eficiente em problemas complexos e com distribuição muito desproporcional de classes.

Comparado a outros métodos na literatura, a metodologia construtiva proposta conseguiu ser significantemente melhor nos problemas Câncer e Tireoide, considerando o erro de classificação no conjunto de teste e o tamanho da arquitetura média das redes. Nos demais problemas, a metodologia obteve resultados satisfatórios, mas não superaram outros métodos encontrados. No entanto, percebe-se outras vantagens inerentes ao modelo proposto, como tamanho reduzido da arquitetura das soluções e tempo necessário para convergência. Como já ressaltado anteriormente, estas duas características agregam qualidade geral ao modelo, pois foi possível obter redes treinadas com boa capacidade de generalização, com arquiteturas pequenas e em um tempo de busca relativamente pequeno. É importante notar no entanto, que o tempo de busca é apenas uma estimativa para referência, pois o algoritmo do modelo não foi implementado com paralelismo real, dado que a ferramenta de desenvolvimento Matlab não dispõe deste recurso de forma explícita. Da mesma forma, o tempo de convergência esta ligado ao hardware e a plataforma utilizada para a execução dos experimentos, divergindo entre as diferentes propostas da literatura.

Em relação à busca manual, a metodologia multimodal proposta se destacou principalmente ao encontrar redes muito menos complexas. Considerando que o método é focado na otimização de pesos e arquiteturas apenas, os resultados demonstram que a otimização é realizada de forma coerente e robusta. Embora não tenha obtido resultados significativamente melhores em todos problemas, a quantidade média de nodos das redes encontradas pela metodologia é evidentemente melhor que a busca manual. O tamanho das arquiteturas encontradas foi menor em todos problemas, destacando que o método proposto cumpre bem a tarefa de busca construtiva de arquiteturas sem comprometer a generalização. Na otimização dos pesos, a metodologia se destacou principalmente no problema da Tireoide, com erros médios de classificação sensivelmente menores que o melhor resultado médio encontrado pela busca manual de arquiteturas.

CAPÍTULO 6

Conclusão

6.1 Considerações Finais

Este trabalho descreveu uma proposta de uma metodologia construtiva e invasiva para otimização multimodal de redes neurais com evolução diferencial. O foco principal do presente trabalho esta contextualizado na integração de estratégias de computação paralela para o desenvolvimento de um sistema neural híbrido uni-modular, simples de ser implementado, mas que obtenha como solução RNAs eficientes em um tempo hábil. Com base nas discussões anteriores, o método proposto obteve resultados promissores, conseguindo até mesmo se igualar ou ser superior a métodos muito mais complexos ou que exploram diversos parâmetros livres na otimização de redes neurais. O bom desempenho é mais evidente no problema do Câncer e Tireoide, cujos resultados apontam uma nítida superioridade a respeito da capacidade de generalização. De modo geral, é possível concluir que a metodologia proposta não ficou limitada a ser eficaz apenas em problemas simples como o do Câncer, mas também em outros tidos como difíceis, onde se inclui o problema da Tireoide.

Além das estratégias utilizadas para a construção do modelo multimodal de otimização de RNAs, este trabalho inova ao comparar a utilização do DE e do ODE como base evolucionária em um sistema neural híbrido. No modelo proposto, o ODE ficou aquém das expectativas, considerando que o algoritmo original DE obteve resultados distintamente melhores. Apesar do ODE ter sido proposto como uma variante que objetiva acelerar a convergência, esta hipótese não pode ser fielmente comprovada na avaliação dos resultados. Neste trabalho, considerou-se o tempo de execução como medida de velocidade de convergência, enquanto os autores originais do ODE utilizaram a quantidade de épocas de otimização. Com esta medida de desempenho, o ODE foi mais lento em todos os experimentos, com exceção do problema da Tireoide. Sugere-se assim que a otimização de RNAs com ODE só apresenta vantagens em relação ao tempo quando utilizada em bases grandes e complexas, permitindo desta forma compensar a complexidade adicional do algoritmo. Entretanto, no âmbito da otimização simultânea de pesos e arquiteturas de redes neurais, conclui-se que estas estratégias para acelerar a convergência inerentes ao ODE ocasionam uma significante perda de qualidade de resposta. De

qualquer forma, pesquisas e resultados com este recente algoritmo ainda encontram-se no limiar da imaturidade científica, tornando este trabalho uma referencia para futuras investigações em outros problemas.

Como discutido nos capítulos anteriores, modelos implícitos de manutenção da diversidade podem, mas não necessariamente garantem que diferentes subpopulações explorem diferentes regiões do espaço de busca. Na comparação com a busca manual de arquiteturas treinadas com DE, os resultados apresentaram coerência com este enunciado, dado que nem sempre a metodologia paralela resultou em uma visível superioridade de qualidade de resposta. Entretanto, o modelo apresentado se destacou visivelmente no problema da Tireoide, tido como o mais difícil da série de experimentos. Não obstante, em todos os problemas o modelo construtivo multimodal obteve redes menos complexas, com uma média de nodos muito inferior às arquiteturas encontradas pela busca manual.

No problema da Íris, a metodologia multimodal obteve erros de classificação ligeiramente superiores que a busca manual. A partir desta evidência, percebe-se uma possível deficiência no critério de parada de migração proposto, dado que que a busca manual conseguiu estes resultados com uma arquitetura distintamente maior. Embora este trabalho apresente uma forma de contornar o frequente problema da parada precipitada inerente aos modelos construtivos, é necessária uma melhor investigação desta estratégia. Sabe-se, contudo, que a busca de arquiteturas desenvolvida neste trabalho é diretamente relacionada à quantidade de subpopulações e a escolha da RNA solução é dependente do critério de parada desenvolvido.

6.2 Trabalhos Futuros

No desenvolvimento deste trabalho, alguns tópicos surgiram como possibilidade de ampliação da proposta original. Devido a área de abrangência dos sistemas híbridos ser vasta, estes tópicos foram incluídos como trabalhos futuros, com o objetivo de aprofundar ou dar continuidade ao presente trabalho.

- Re-implementação do método em ferramentas de baixo nível com paralelismo real. Para a realização dos experimentos, a metodologia foi implementada utilizando Matlab, ferramenta de programação em alto nível e reconhecidamente lenta, além de não permitir paralelismo explícito de processamento. O desenvolvimento da metodologia proposta em uma linguagem mais eficiente e que permita paralelismo permitiria uma melhor avaliação do tempo de processamento de busca, reafirmando ou melhorando esta importante característica do método.

- Propostas de novos critérios de parada. A metodologia multimodal possui basicamente dois critérios de parada inerentes ao algoritmo: para a migração e busca de arquiteturas; para o ajuste dos pesos da RNA. Como pôde ser observado na comparação com busca manual, o critério de parada migração ainda pode estar limitando a busca por arquiteturas em alguns problemas, não propondo uma quantidade minimamente razoável de nodos para uma solução objetivo. Logo, o critério de parada da migração deve ser melhor investigado para manter uma coerência com a busca multimodal de arquiteturas.

- Experimentos com novos tipos de problemas. Neste trabalho, a série de experimentos limitou-se a problemas bem conhecidos de classificação. Os testes realizados serviram para validar a boa performance do modelo, contudo, não foram exaustivos. Como extensão, seria interessante avaliar os resultados utilizando problemas de outros domínios, como os de previsão de séries temporais, considerados de difícil solução.

- Utilizar outras bases evolucionárias. Este trabalho propõe uma metodologia que integra estratégias para sobre-otimizar algoritmos evolucionários na composição de um Sistema Neural Híbrido. Dessa forma, este método pode ser facilmente generalizado para utilização de outros algoritmos baseados em população, como Algoritmos Genéticos, Otimização por Enxame de Partículas, Estratégias de Evolução, etc.

- Otimização de outras topologias de RNAs. Uma das principais limitações deste trabalho, refere-se ao SNH proposto estar limitado a redes neurais com apenas uma camada oculta. Dessa forma, seria de grande valia modificar a metodologia para que incluí-se uma relação de crescimento estrutural considerando duas ou mais camadas ocultas, tarefa difícil em algoritmos construtivos que consideram RNAs com nodos distribuídos em camadas e com apenas conexões adjacentes. Ainda neste escopo, a metodologia poderia ser igualmente adaptada e avaliada na otimização de outras topologias de RNAs, como redes com atraso de tempo ou recorrência interna.

Referências Bibliográficas

[Abb02] H.A. Abbass. An evolutionary artificial neural networks approach for breast cancer diagnosis. *Artificial Intelligence in Medicine*, 25(3):265–281, 2002.

[Abb03] H.A. Abbass. Pareto neuro-evolution: constructing ensemble of neural networks using multi-objective optimization. In *IEEE Congress on Evolutionary Computation, 2003. CEC 2003*, volume 3, pages 2074–2080, 2003.

[Abr04] A. Abraham. Meta-learning evolutionary artificial neural networks. *Journal, Elsevier Science, Netherlands*, 56:1–38, 2004.

[AK09] H.M. Abdul-Kader. Neural networks training based on differential evolution algorithm compared with other architectures for weather forecasting34. *Internation Journal of Computer Science and Network Security*, 9(3):92–99, 2009.

[AL06] L.M. Almeida and T.B. Ludermir. A hybrid method for searching near-optimal artificial neural networks. In *6th International Conference on Hybrid Intelligent Systems, 2006. HIS 2006*, pages 36–36, 2006.

[AL08] L.M. Almeida and T. Ludermir. An improved method for automatically searching near-optimal artificial neural networks. In *IEEE International Joint Conference on Neural Networks, 2008. IJCNN 2008*, pages 2235–2242, 2008.

[AN00] A. Abraham and B. Nath. Hybrid intelligent systems design - a review of a decade of research. Technical report, Monash University, 2000.

[AN07] A. Asuncion and D.J. Newman. UCI machine learning repository [http://www.ics.uci.edu/∼mlearn/MLRepository.html], 2007. Acessado em 1 de julho de 2009.

[ASP94] P.J. Angeline, G.M. Saunders, and J.B. Pollack. An evolutionary algorithm that constructs recurrent neural networks. *IEEE Transactions on Neural Networks*, 5(1):54–65, 1994.

[BCL07] A.P. Braga, A.P.L.F. Carvalho, and T.B. Ludermir. *Redes Neurais Artificiais: Teorias e Aplicações*. LTC - Livros Técnicos e Científicos, 2007.

[Bhu09] M.Z.A. Bhuiyan. An algorithm for determining neural network architecture using differential evolution. In *International Conference on Business Intelligence and Financial Engineering, 2009. BIFE 2009*, pages 3–7, 2009.

[Bit06] G. Bittencourt. *Inteligência Artificial - Ferramentas e Teorias*. Editora da UFSC, 2 edition, 2006.

[BR05] P.K. Bergey and C. Ragsdale. Modified differential evolution: a greedy random strategy for genetic recombination. *Omega*, 33(3):255–265, 2005.

[Cha08] U.K. Chakraborty. *Advances in Differential Evolution*. Springer Publishing Company, Incorporated, 2008.

[CL07] M. Carvalho and T.B. Ludermir. Particle swarm optimization of neural network architectures and weights. In *7th International Conference on Hybrid Intelligent Systems, 2007. HIS 2007*, pages 336–339, 2007.

[ES03] A. Eiben and J. Smith. *Introduction to evolutionary computing*. Springer, 2003.

[GA08] M.M. Gouvae and A. Araujo. Diversity control based on population heterozygosity dynamics. In *IEEE Congress on Evolutionary Computation, 2008. CEC 2008*, pages 3671–3678, 2008.

[GT92] M. Gori and A. Tesi. On the problem of local minima in backpropagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(1):76–86, 1992.

[GTT08] Chi-Keong Goh, Eu-Jin Teoh, and Kay Chen Tan. Hybrid multiobjective evolutionary design for artificial neural networks. *IEEE Transactions on Neural Networks*, 19(9):1531–1548, 2008.

[Hay98] S. Haykin. *Neural Networks: A Comprehensive Foundation (2nd Edition)*. Prentice Hall, 2 edition, 1998.

[Heb49] D.O. Hebb. *The Organization of Behavior: A Neuropsychological Theory*. Wiley, New York, new edition, 1949.

[Hol75] J.H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, USA, 1975.

[IKL03] J. Ilonen, J.K. Kamarainen, and J. Lampinen. Differential evolution training algorithm for feed-forward neural networks. *Neural Processing Letters*, 17(1):93–105, 2003.

[IM01] Md.M. Islam and K. Murase. A new algorithm to design compact two-hidden-layer artificial neural networks. *Neural Networks*, 14(9):1265–1278, 2001.

[KE95] J. Kennedy and R. Eberhart. Particle swarm optimization. In *IEEE International Conference on Neural Networks, 1995*, volume 4, pages 1942–1948, 1995.

[LV91] J. Lin and J.S. Vitter. Complexity results on learning by neural nets. In *Machine Learning*, pages 211–230, 1991.

[LYZ06] T.B. Ludermir, A. Yamazaki, and C. Zanchettin. An optimization methodology for neural network weights and architectures. *IEEE Transactions on Neural Networks*, 17(6):1452–1459, 2006.

[MK04] L. Ma and K. Khorasani. New training strategies for constructive neural networks with application to regression problems. *Neural Networks*, 17(4):589–609, 2004.

[ML97] T. Masters and W. Land. A new training algorithm for the general regression neural network. In *IEEE International Conference on Systems, Man, and Cybernetics, 1997. 'Computational Cybernetics and Simulation' 1997*, volume 3, pages 1990–1994, 1997.

[MP43] W. Mcculloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biology*, 5(4):115–133, 1943.

[NGN05] P. Neto, L. Garcia, and M.C. Nicoletti. *Introdução às redes neurais construtivas*. EdUFSCAR, 2005.

[PHU05] P.P. Palmes, T. Hayasaka, and S. Usui. Mutation-based genetic neural network. *IEEE Transactions on Neural Networks*, 16(3):587–600, 2005.

[PMB05] N.G. Pedrajas, C.H. Martinez, and D.O. Boyer. Cooperative coevolution of artificial neural network ensembles for pattern classification. *IEEE Transactions on Evolutionary Computation*, 9(3):271–302, 2005.

[Pre94] L. Prechelt. PROBEN1 — A set of benchmarks and benchmarking rules for neural network training algorithms. Technical Report 21/94, Fakultät für Informatik, Universität Karlsruhe, 1994. Anonymous FTP: /pub/papers/techreports/1994/1994-21.ps.Z on ftp.ira.uka.de. Acessado em 1 de julho de 2009.

[PSL05] K.V. Price, R.M. Storn, and J.A. Lampinen. *Differential Evolution A Practical Approach to Global Optimization*. Natural Computing Series. Springer-Verlag, Berlin, Germany, 2005.

[RHW86] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. *Parallel distributed processing: explorations in the microstructure of cognition, vol. 1: foundations*, pages 318–362, 1986.

[Ros58] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

[RTS08] S. Rahnamayan, H.R. Tizhoosh, and M.M.A. Salama. Opposition-based differential evolution. *IEEE Transactions on Evolutionary Computation*, 12(1):64–79, 2008.

[SA99] G.P.J. Schmitz and C. Aldrich. Combinatorial evolution of regression nodes in feedforward neural networks. *Neural Networks*, 12(1):175–189, 1999.

[SB08] A. Slowik and M. Bialko. Training of artificial neural networks using differential evolution algorithm. In *Conference on Human System Interactions, 2008*, pages 60–65, 2008.

[SP95] R. Storn and K. Price. Differential evolution – a simple efficient adaptive scheme for global optimization over continuous spaces. Technical Report 95-012, Int. Compt. Sci. Inst., Berkeley, CA, 1995.

[Tem09] F. Temurtas. A comparative study on thyroid disease diagnosis using neural networks. *Expert Systems with Applications*, 36(1):944–949, 2009.

[Tiz05] H.R. Tizhoosh. Opposition-based learning: A new scheme for machine intelligence. In *International Conference on Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, 2005*, volume 1, pages 695–701, 2005.

[TPPV04] D.K. Tasoulis, N.G. Pavlidis, V.P. Plagianakos, and M.N. Vrahatis. Parallel differential evolution. In *IEEE Congress on Evolutionary Computation, 2004. CEC 2004*, volume 2, pages 2023–2029, 2004.

[TS02] T. Takahama and S. Sakai. Structural optimization of neural network by genetic algorithm with damaged genes. In *9th International Conference on Neural Information Processing, 2002. ICONIP 2002*, volume 3, pages 1211–1215, 2002.

[TSHI07] T. Takahama, S. Sakai, A. Hara, and N. Iwane. Structural learning of neural networks by differential evolution with degeneration using mappings. In *IEEE Congress on Evolutionary Computation, 2007. CEC 2007*, pages 3434–3441, 2007.

[TSI02] T. Takahama, S. Sakai, and Y. Isomichi. Mgga: Genetic algorithm with mutant genes. *Systems and Computers in Japan*, 33(14):23–33, 2002.

[VT04] J. Vesterstrom and R. Thomsen. A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. In *IEEE Congress on Evolutionary Computation, 2004. CEC 2004*, volume 2, pages 1980–1987, 2004.

[WRH90] A.S. Weigend, D.E. Rumelhart, and B.A. Huberman. Generalization by weight-elimination with application to forecasting. In *Advances in neural information processing systems 3*, pages 875–882, 1990.

[Yao99] X. Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, 1999.

[yKY97] Tin yau Kwok and Dit-Yan Yeung. Constructive algorithms for structure learning in feedforward neural networks for regression problems. *IEEE Transactions on Neural Networks*, 8:630–645, 1997.

[ZL07] C. Zanchettin and T.B. Ludermir. Global optimization methods for designing and training feedforward artificial neural networks. *Dynamics of Continuous, Discrete and Impulsive System*, 14:328–337, 2007.

[ZL09] A.M.F. Zarth and T.B. Ludermir. Optimization of neural networks weights and architecture: A multimodal methodology. In *Ninth International Conference on Intelligent Systems Design and Applications, 2009. ISDA 2009*, pages 209–214, 2009.