Métodos de Otimização Global para Escolha do Padrão de Conectividade de Redes Neurais sem Peso

Dissertação de Mestrado

Por Luiz Alberto Crispiniano Garcia

Orientador: Prof. Dr. Marcílio Carlos Pereira de Souto

Recife

Março de 2003

Universidade Federal de Pernambuco
Centro de Informática
Mestrado em Ciência da Computação

Métodos de Otimização Global para Escolha do Padrão de Conectividade de Redes Neurais sem Peso

Dissertação submetida à Universidade Federal de Pernambuco para o grau de Mestre em Ciência da Computação na área de Computação Inteligente

Por Luiz Alberto Crispiniano Garcia

Orientador: Prof. Dr. Marcílio Carlos Pereira de Souto

Recife
Março de 2003

Aos meus pais e à Raíssa pela dedicação, amor, carinho e apoio.

Resumo

Nas Redes Neurais Sem Peso (RNSP), o padrão de conectividade dos nós desempenha um papel fundamental, pois é diretamente responsável pela performance da rede, determinando o custo computacional, a eficiência, a velocidade, a tolerância a falhas e ruídos e a capacidade de generalização. Porém, apesar de sua importância, de um modo geral, o padrão de conectividade é escolhido empiricamente, por meio de um método manual de tentativas e erros ou com a ajuda de um especialista, o que nem sempre é viável.

Por outro lado, essa escolha do melhor padrão de conectividade pode ser visto como um problema de otimização, no qual cada rede é um ponto no espaço de soluções possíveis. Portanto, vários métodos de otimização, principalmente os métodos globais de otimização, têm sido propostos para automatização da geração do padrão de conectividade. Esses métodos fazem busca no espaço global, evitando cair em soluções de mínimos locais.

Nesse contexto, uma das principais contribuições desta dissertação é um estudo experimental sobre o uso de métodos de otimização global, tais como Algoritmos Genéticos, *Simulated Annealing* e *Tabu Search*, aplicados à escolha do padrão de conectividade das RNSPs. Os resultados obtidos com esses métodos, junto com aqueles baseados na escolha empírica, são avaliados no contexto de um problema de classificação de caracteres numéricos manuscritos, em que testes de hipótese são aplicados.

Por exemplo, o uso de *Tabu Search* conseguiu diminuir em 17,27% o erro médio de classificação obtido inicialmente com o método de escolha empírica. Em outro experimento, o uso de algoritmos genéticos conseguiu diminuir em 89% o uso de memória alocada, com erros médios de classificação menores que os obtidos inicialmente, pelo método baseado na escolha empírica.

Abstract

In Weightless Neural Networks (WNN), the connectivity pattern of the nodes in a network is of fundamental importance, because such a pattern reflects on the performance of the network, its computational cost, efficiency, ability to deal with fault and noise, and generalization capacity. However, despite the importance of choosing the connectivity pattern, this choice is often accomplished empirically applying a trial and error method or with expertise help, which it is not always available.

On the other hand, this choice of the best connectivity pattern can be seen as an optimization problem, where each neural network is regarded as a point in the search space. Therefore, several optimization methods, mainly the global ones, have been proposed to implement the process of generating the connectivity pattern. These methods search in the global space, avoiding local minimum solutions.

In this context, this dissertation proposes an experimental study on the use of global optimization methods, such as Genetic Algorithm, Simulated Annealing and Tabu Search, applied to choose the connectivity pattern of WNNs. The results obtained with these methods, together with those achieved with trial-error method, are evaluated based on hypothesis tests in the context of a classification problem (classification of hand-written numerical characters).

For example, in an experiment, the use of Tabu Search decreased in 17.27% the mean of the classification errors of the networks, when compared to those obtained by the trial-error method. In other experiment, the application of Genetic Algorithms not only decreased in 89% the use of memory, but also the mean of the classification errors obtained were lower than the ones initially achieved by the trial-error method.

Agradecimentos

Registrar em apenas uma página, os agradecimentos a todos aqueles que contribuíram, de forma direta ou indireta, para o desenvolvimento deste trabalho, não é uma tarefa fácil. Por isso, desejo agradecer de coração não só aos citados aqui mas a todos que participaram comigo dessa caminhada.

Em primeiro lugar, minha eterna gratidão à minha família – meus pais, minha amada Raíssa e meus irmãos – por todo o apoio, carinho, compreensão, motivação, calma e amor, antes, durante e depois desse período. Em particular, à minha mãe que teria ficado muito orgulhosa, se estivesse aqui. Peço também desculpas a vocês pelas ausências, pelo cansaço e pelo estresse.

Gostaria, também, de agradecer aos professores do Centro de Informática, em especial, ao professor Fernando Fonseca, que se revelou excelente professor de programação e conselheiro, e ao professor Augusto César que se tornou um amigo. Mas a área que sempre sonhei, Inteligência Artificial, me foi apresentada em 1993 pela professora Teresa Bernarda, orientadora e conselheira, sendo ela crucial na descoberta de meu sonho profissional, que é a pesquisa em Redes Neurais Artificiais.

Ao Serpro, a todos os meus amigos e companheiros de trabalho que me permitiram cursar o Mestrado enquanto trabalhava, servindo como fonte inesgotável de aprendizado. Em especial, gostaria de agradecer à Helena Cristina, gerente da minha área, e a minha chefe e amiga, Mônica Falcão, que sempre me apoiou e me incentivou.

Gostaria de agradecer àqueles que me ajudaram nesse longo período, amigos como Marcelo Oliveira, Leonardo França, Akio Yamazaki e George Darmiton. A Anne Magali, pelo apoio e pelas bases manuscritas.

Agradeço, por último, e, de modo especial, ao meu orientador, professor Marcílio, pelo apoio, pelos conselhos, pelo empenho, pela confiança em mim e pelas noites de conversas e correções. Sem ele, teria sido impensável ter concluído meu Mestrado.

Índice

Lista de Siglas

AG	Algoritmo Genético
EC	Erro de Classificação
PLN	*Probabilistic Logic Node*
RAM	*Random Access Memory*
RNA	Rede Neural Artificial
RNSP	Rede Neural Sem Peso
SA	*Simulated Annealing*
TS	*Tabu Search*

Capítulo 1

1 Introdução

1.1 Motivação

Redes Neurais Artificiais (RNAs) vêm sendo utilizadas, por exemplo, na solução de problemas, como o controle de segurança por meio do reconhecimento de assinaturas [LEE et al., 1997], impressões digitais [PRECISEBIOMETRICS], íris [IRIDIANTECH]; para o reconhecimento de safras de vinhos [YAMAZAKI et al., 2002]; para a compressão de imagens [WATTA et al., 1996]. As RNAs apresentam soluções para esses problemas que anteriormente necessitariam de especialistas que demandariam tempo e ofereceriam soluções, muitas vezes, subjetivas e, às vezes, incorretas.

As RNAs podem resolver alguns problemas, embora também produzam outros. Um deles é a definição de uma boa arquitetura para uma determinada tarefa. De uma forma geral, RNAs são formadas por uma rede de neurônios artificiais, que possuem uma estrutura bastante simples, embora sejam capazes de apresentar um comportamento complexo quando são conectados entre si [BRAGA et al., 2000]. A arquitetura da rede (tamanho, estrutura e conexões) é diretamente determinante de sua performance na resolução do problema. Portanto, uma escolha mal feita da arquitetura pode implicar um mau desempenho da rede.

Em geral, esse problema é abordado por meio da geração de várias arquiteturas – um processo de tentativa e erro [PRECHELT, 1994] – no qual se escolhe a melhor, de acordo com alguma medida de desempenho para um dado problema. Por fim, a melhor rede da arquitetura escolhida é, então, usada para responder ao dado problema. Nesta dissertação, esse método será denominado de *método tradicional*. Apesar do uso generalizado do método tradicional, esse método pode não apresentar resultados confiáveis, pois não há garantias da obtenção de uma arquitetura com o desempenho ótimo, caso ela exista, ou mesmo, subótimo.

Como o problema de escolha de arquitetura é muito relevante, pesquisadores têm sugerido meios de melhorar o desempenho das arquiteturas selecionadas, por exemplo, usando técnicas de otimização global [SEXTON et al., 1998] [YAO, 1999] [YAMAZAKI et al., 2002] que podem modificar, entre outras mudanças possíveis, o padrão de conectividade dos neurônios e o número de neurônios da arquitetura.

Nesse contexto, algoritmos genéticos vêm apresentando bons resultados [YAO, 1999] e, apenas recentemente, outros métodos, como *simulated annealing* e *tabu search*, vêm sendo usados [BISHOP et al., 1990] [YAMAZAKI et al., 2002]. Sendo assim, um trabalho que compare os resultados obtidos, a partir de arquiteturas otimizadas pelos três métodos, algoritmos genéticos, *simulated annealing* e *tabu search*, com o método tradicional, para um mesmo problema, seria uma importante contribuição.

1.2 Objetivos e Abordagem

O principal objetivo deste trabalho é o de desenvolver uma análise comparativa do método tradicional (tentativa e erro) e do uso de três métodos de otimização global, algoritmos genéticos, *simulated annealing* e *tabu search*, para otimizar a topologia de RNAs.

Mais especificamente, o estudo tem seu foco nos métodos de otimização global, usados para otimizar o padrão de conectividade de redes neurais sem peso [LUDERMIR et al., 1999], a fim de investigar o desempenho destas em tarefas de classificação. A análise do desempenho dos

métodos é baseada em um problema do mundo real (o reconhecimento de caracteres numéricos manuscritos) e não em problemas simples, normalmente utilizados na literatura [BISHOP et al., 1990] [CHALUP e MAIRE, 1999] [STORK e ALLEN, 1992].

Outro objetivo desta dissertação é o de sugerir uma forma de codificar as redes neurais sem peso, já que até o momento, no contexto dessas redes, não foi encontrada nenhuma forma de codificá-las. Essa codificação é necessária, pois os métodos de otimização global, de forma geral, não trabalham diretamente com as possíveis soluções do problema, mas, com uma representação indireta destas. Essa representação permite armazenar o padrão de conectividades e manipulá-lo, de forma a gerar novos padrões de conectividades possíveis.

Duas bases de dados distintas são usadas para uma melhor investigação de como se comportam os métodos de otimização global. Para cada base de dados, dezesseis topologias diferentes de redes neurais são aplicadas na tarefa de reconhecimento de caracteres numéricos manuscritos. Em cada topologia, trinta redes diferentes são geradas pelo método tradicional. Essa quantidade foi escolhida, para poder se obter resultados estatísticos mais robustos em relação a médias e desvios padrões.

Após as dezesseis topologias terem sido aplicadas ao problema de reconhecimento, três delas são escolhidas e otimizadas pelos métodos de algoritmos genéticos, *simulated annealing* e *tabu search*. Por fim, os erros de classificação, obtidos pelos métodos, são comparados, utilizando-se uma técnica estatística de teste de hipótese, para verificar se as diferenças entre erros médios de classificação dos métodos são significantes ou acidentais.

Todos os experimentos são implementados em C#.Net©, e todos os dados (as duas bases de dados e as redes neurais sem peso obtidas pelos métodos) são armazenados em um servidor de banco de dados MS SQL Server 2000©. Os erros de classificação obtidos são organizados em planilhas MS Excel©, nas quais são realizados os testes estatísticos de hipóteses.

1.3 Principais Contribuições

A primeira contribuição deste trabalho é um estudo sobre o resultado obtido com a integração de métodos de otimização global com redes neurais sem peso, usando um problema do mundo real, um exemplo do uso de um sistema inteligente híbrido. [LUDERMIR et al., 2003]

Outra contribuição deste trabalho é a de apresentar resultados da aplicação de um modelo de redes neurais sem peso para o reconhecimento de caracteres numéricos manuscritos com alguns comentários relevantes. A terceira contribuição é a análise comparativa dos resultados para verificar se as diferenças entre os métodos são significantes ou acidentais, realizada através de um método estatístico de teste de hipótese.

A forma de representação das redes neurais sem peso também servirá de contribuição para outros trabalhos futuros. Também poderão ser usadas, em trabalhos posteriores, as implementações feitas para criar classificadores e experimentos de otimização. E, por fim, este trabalho servirá de material introdutório para novos trabalhos, pois, no contexto dos modelos neurais sem peso, não foi encontrado nenhum trabalho equivalente na literatura.

1.4 Estrutura da Dissertação

O restante desta dissertação está dividido em quatro capítulos:

Capítulo 2

Nesse capítulo, é apresentado um modelo de redes neurais sem peso, utilizado para classificação bem como os três métodos, algoritmos genéticos, *simulated annealing* e *tabu search*. Logo após, é feito um resumo geral dos trabalhos relevantes encontrados na literatura. Ao final do capítulo, este trabalho é posto em perspectiva em relação a outros estudos encontrados na literatura.

Capítulo 3

Nesse capítulo, é feita uma descrição do problema, das bases usadas com os particionamentos e de como foram desenvolvidos os experimentos, acompanhados dos devidos resultados resumidos, tabelados e comentados. Também é apresentado o método estatístico de teste de hipótese, usado para comparar os resultados.

Capítulo 4

Nesse capítulo, é apresentado um comparativo entre os resultados obtidos com os métodos, indicando quais deles apresentaram diferenças significativas, sendo levantadas algumas considerações.

Capítulo 5

Nesse último capítulo, é feito um resumo do contexto do trabalho desenvolvido e dos objetivos alcançados, junto com algumas conclusões e sugestões para trabalhos futuros.

Capítulo 2

2 Revisão Bibliográfica

2.1 Introdução

Neste capítulo, a fim de propiciar a base necessária para o entendimento do Capítulo 3, é feita uma breve revisão de assuntos pertinentes a este trabalho. Na Seção 2.2, dois modelos neurais são apresentados, o modelo neural com peso e o modelo neural sem peso, com foco maior nesse último, pois é o modelo usado neste trabalho para classificação de padrões [BRAGA et al., 2000] [SOUTO, 1999]. Na Seção 2.3, os três métodos de otimização global, usados neste trabalho, são apresentados: algoritmos genéticos, *simulated annealing* e *tabu search*. Na Seção 2.4, é descrito, em linhas gerais, o método de treinamento híbrido proposto por este trabalho. Na Seção 2.5, é apresentado um resumo dos trabalhos relacionados, encontrados na literatura. Ao final, na Seção 2.6, o trabalho proposto nesta dissertação será posto em perspectiva em relação aos trabalhos revisados.

2.2 Modelos Neurais Artificiais

Os modelos neurais, usados na maioria dos trabalhos que envolvem redes neurais, são variações de neurônios McCulloch-Pitts (MCP) [MCCULLOCH e PITTS, 1943], que serão denominados, nesta dissertação, de *neurônios com peso*. Um neurônio com peso típico (Figura 2.1) pode ser descrito pelas Equações 2.1 e 2.2 que especificam uma soma ponderada das entradas e alguma função de transferência não linear,

$$h = \sum_{j=1}^{p} W_j u_j \tag{2.1}$$

$$y = \frac{1}{1 + \exp^{(-ah)}} \tag{2.2}$$

em que, u_j representa as entradas; W_j, os pesos sinápticos dos neurônios; p, o tamanho da entrada; h, a *ativação* do neurônio; a, um parâmetro de controle da forma da função sigmoidal; e y, a saída do neurônio. A função y acima pode ser substituída por qualquer outra função, por exemplo, uma função limiar. Entretanto, o uso de funções sigmóides é comum pela necessidade de ter funções de erro que sejam diferenciáveis em relação aos pesos, permitindo que algoritmos de gradiente descendente, como o algoritmo de *backpropagation* [RUMELHART et al., 1986], possam ser usados no aprendizado. Os modelos neurais compostos por neurônios com peso serão chamados nesta dissertação de *redes neurais com peso*.

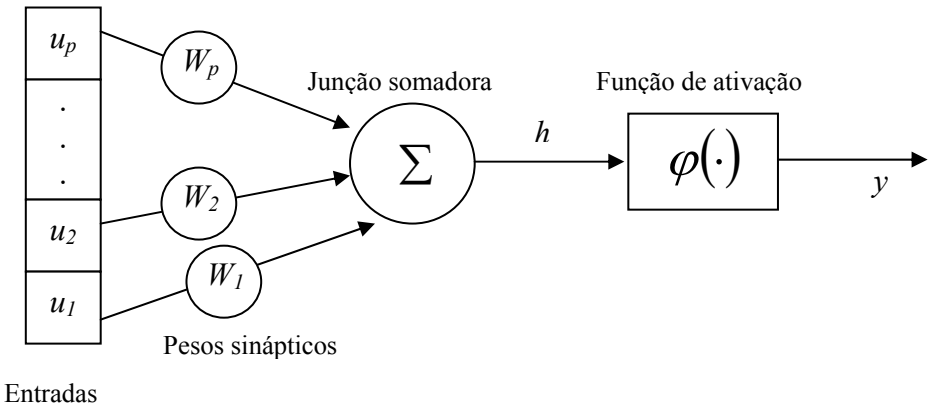


Figura 2.1 - Um neurônio com peso típico.

Por outro lado, o modelo de computação neural, usado neste trabalho, é baseado em neurônios artificiais que têm entradas e saídas binárias e nenhum peso entre os nós. As funções dos neurônios desses modelos são armazenadas em tabelas-verdade que podem ser implementadas, utilizando memórias de acesso aleatório (*Random Access Memories* - RAM) comuns [LUDERMIR et al., 1999]. Na literatura, termos, como *modelos neurais sem peso*, "*RAM-based*" ou "*n-tuple based*", são usados para se referir a esses modelos.

No contexto dos modelos neurais sem peso, ensinar uma rede consiste apenas em modificar conteúdos de memórias dos neurônios. Isso resulta em um algoritmo de aprendizado mais simples e rápido do que os algoritmos usados nas redes com peso, em que ensinar ocorre com o ajuste de pesos entre os neurônios utilizando gradiente descendente [ALEKSANDER e STONHAM, 1979] [BLEDSOE e BROWNING, 1959] [BRAGA et al., 2000]. Além disso, ao aprender um novo padrão de entrada e saída nas redes com peso, o comportamento da rede em relação a padrões, aprendidos previamente, pode mudar, tornando o processo de aprendizado muito mais complexo. Nas redes neurais sem peso, usadas nesta dissertação, um novo padrão aprendido não altera, significativamente, o comportamento da rede para outros padrões aprendidos previamente, sendo necessário mostrar o conjunto de treinamento apenas uma vez, usando um algoritmo do tipo *one-shot-learning* [ALEKSANDER e MORTON, 1991] para o aprendizado.

O modelo neural sem peso se originou com as máquinas de amostragem de *n*-tupla, descritas por Bledsoe e Browning [BLEDSOE e BROWNING, 1959], nos anos 50. Logo após, Aleksander [ALEKSANDER, 1966] interessou-se no aprendizado adaptativo em redes neurais, usando máquinas de amostragem de *n*-tupla, descritas por Bledsoe e Browning; ele sugeriu um circuito lógico universal, como o nodo de uma rede que aprende. Ele também introduziu os nodos RAM e SLAM (*Stored Logic Adaptive Microcircuit*), que foram projetados com o propósito de pesquisa antes do advento de memória de circuitos integrados.

Ao final dos anos 70, a construção de grandes redes neurais, que permitissem lidar com imagens de alta qualidade se tornou possível, porque os nodos RAM eram muito baratos e com alta capacidade. A construção do WISARD (*Wilkie, Stonham and Aleksander's Recognition Device* ou dispositivo de reconhecimento de Wilkie, Stonham e Aleksander) ficou viável, sendo que o primeiro protótipo foi concluído em 1981 [ALEKSANDER et al., 1984]. A máquina foi

patenteada e produzida comercialmente desde 1984. Após o WISARD, outros modelos sem peso foram propostos na literatura, como o PLN (*Probabilistic Logic Node*) [KAN e ALEKSANDER, 1987], o MPLN (*Multiple-valued Probabilistic Logic Node*) [MYERS e ALEKSANDER, 1988] [MYERS e ALEKSANDER, 1989], o pRAM (*probabilistic* RAM) [GORSE e TAYLOR, 1990] [GORSE e TAYLOR, 1990] e o GRAM (*generalising* RAM) [ALEKSANDER, 1989] .

As principais vantagens das redes neurais sem peso sobre as redes neurais com peso são o aprendizado rápido, a facilidade de implementar as redes em paralelo e a possibilidade de implementar redes reais em hardware, usando memórias RAM. As principais desvantagens de um neurônio sem peso em relação ao com peso são: o tamanho da memória que cresce exponencialmente com o tamanho da quantidade de conexões, impedindo-os de serem completamente conectados e tornando-os parcialmente conectados [BRAGA et al., 2000]; o neurônio RAM descrito abaixo não consegue generalizar, isso acontece apenas quando se forma uma rede com vários desses neurônios, tornando-os dependentes de seu padrão de conectividade. [ALEKSANDER, 1983]

2.2.1 Neurônio RAM

Um neurônio RAM funciona no domínio discreto, recebendo como entrada e produzindo como saída um valor binário. O neurônio RAM é capaz de computar qualquer função lógica com um certo número de entradas, enquanto que neurônios com peso computam apenas funções linearmente separáveis. Seu nome decorre da analogia que é feita com as memórias RAM [BRAGA et al., 2000]. Formalmente, ele apresenta a seguinte definição:

Definição 2.2.1 [Souto, 1999] Um neurônio RAM (Figura 2.2) é um neurônio artificial com as seguintes subdefinições e restrições:

- $E = \{0, 1\}^{n_I}$ é o conjunto de entradas do neurônio, sendo n_I a quantidade de conexões, também chamado de *fan-in*.
- $K = \{0, 1\}^{n_I}$ é o conjunto de endereços do neurônio. Para cada endereço $k \in K$, existe uma célula *C[k]* que armazena o conteúdo ou informação aprendida (memória local) na

forma de um bit. Um sinal binário $e \in E$ no terminal de entrada irá acessar apenas a célula $k=e$. O conteúdo k acessado é chamado de conteúdo ativado.

- $s \in S$ é a saída do neurônio, sendo S o conjunto $\{0, 1\}$.
- $r \in S$ é o terminal de treinamento com a resposta desejada.
- $m=\{0, 1\}$ é o terminal de modo de operação, indicando se o neurônio está na fase de aprendizado ou uso.
- $f:E \rightarrow S$ é a *função de transferência* que computa s a partir do bit armazenado no endereço, determinado pelo terminal de entrada, sendo $s = f(C[k = e])$.

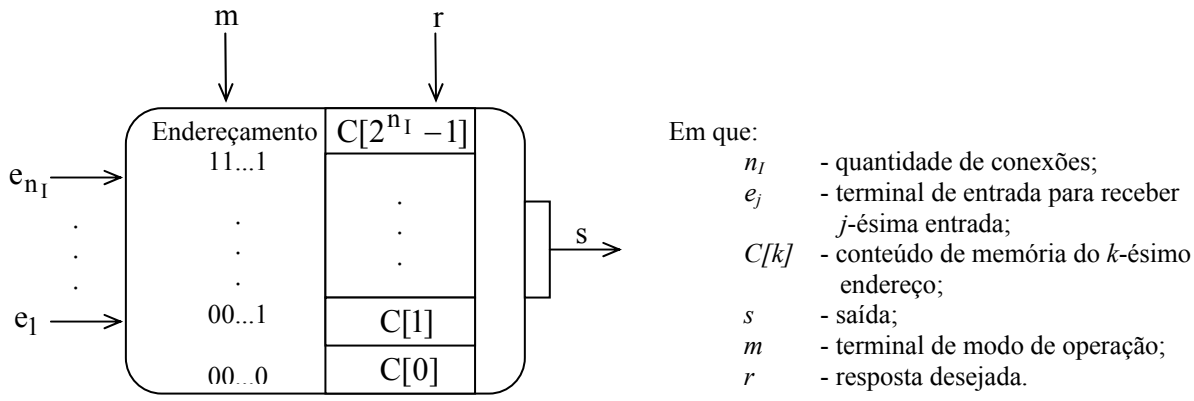


Figura 2.2 - Exemplo de um neurônio RAM

Ensinar um neurônio RAM consiste, apenas, em modificar o conteúdo de tabelas verdades. Esta simplicidade torna essas redes bastante rápidas tanto na aprendizagem quanto no uso. Além disso, como suas entradas e saídas são binárias, elas são facilmente implementadas em hardware. Como o tamanho da memória de um neurônio RAM cresce exponencialmente com o tamanho do *fan-in* (quantidade de conexões de entrada), esses neurônios são, geralmente, parcialmente conectados. Dessa forma, é preciso decidir um tamanho de *fan-in* ideal para o problema. Ele não deve ser muito grande, pois, ao crescer exponencialmente a memória, o neurônio acaba se especializando, apenas, nas características aprendidas no conjunto de treinamento. Assim, ele necessitará de uma maior variabilidade no conjunto de treinamento para melhorar seu desempenho. O *fan-in* também não deve ser muito pequeno, a fim de minimizar o problema de saturação no armazenamento dos conteúdos nas memórias.

Cada neurônio reconhece, apenas, padrões a que ele foi submetido na fase de treinamento, não havendo generalização por parte do neurônio propriamente dito. A generalização é obtida, ao combinar as subcaracterísticas que cada neurônio individualmente consegue reconhecer em um padrão [ALEKSANDER, 1983] [BRAGA et al., 2000].

Por fim, as memórias dos neurônios RAM são normalmente inicializadas com 0. Nesse caso, embora as posições de memória contendo 1 não possuam informações ambíguas, o mesmo não pode ser dito das posições contendo 0. Um padrão de entrada que acesse uma posição que contenha 0 tanto pode significar um contra-exemplo da classe quanto um padrão da classe cujas características não foram ensinadas pelos padrões de treinamento da classe. Para resolver esse problema de ambigüidade de posições de memória contendo 0 dos neurônios RAM, foi desenvolvido o neurônio PLN (*Probabilistic Logic Node*) [KAN e ALEKSANDER, 1987].

2.2.2 Redes RAM

Uma típica rede RAM é uma rede *feedforward* (um grafo acíclico) com apenas uma camada de neurônios [SOUTO, 1999]. Os neurônios RAM são inicializados com 0 em todas as posições de memória e são ensinados a responder com 1 para os padrões no conjunto de treinamento. Um padrão ainda não apresentado anteriormente à rede é classificado como sendo da mesma classe do conjunto de treinamento, se todos os neurônios responderem com 1. Essa arquitetura simples divide o conjunto de todos os padrões em padrões que estão e os que não estão dentro do conjunto generalizado dos padrões de treinamento. [BRAGA et al., 2000]

Quando mais de uma característica ou classe é necessária para ser discriminada, combinam-se várias redes RAM, em que cada rede é responsável por discriminar uma classe. A decisão, à qual categoria pertence o padrão novo apresentado às redes classificadoras, é dada por um detector de resposta máxima. Cada rede é chamada de discriminador, e o conjunto de discriminadores é denominado de multidiscriminador. [BRAGA et al., 2000] [SOUTO, 1999]

Melhor dizendo, um *discriminador* (Figura 2.3) consiste em uma rede de uma única camada com k neurônios RAM de n entradas cada. Cada neurônio é conectado a uma parte do padrão de entrada, aprendendo, apenas, parte dele na fase de treinamento. A saída de um discriminador

para um padrão, na fase de teste, é um número binário com *k* bits que é processado por um dispositivo somador, dando como saída o número de neurônios RAM que dispararam para aquele padrão de entrada. Já um *multidiscriminador* consiste em diversos discriminadores combinados, sendo cada um treinado para o reconhecimento de uma classe de padrões (Figura 2.4). No caso deste trabalho, todo multidiscriminador possui dez discriminadores, sendo um discriminador para cada classe de caractere numérico.

O tipo de algoritmo de treinamento *one-shot-learning* usado neste trabalho segue os seguintes procedimentos [LUDERMIR et al., 1999]: durante a fase de treinamento de um multidiscriminador, para cada padrão *p* do conjunto de treinamento, ocorre a gravação do padrão, apenas, nos neurônios RAM, que correspondem ao discriminador D_i, responsável pela classe do padrão *p*.

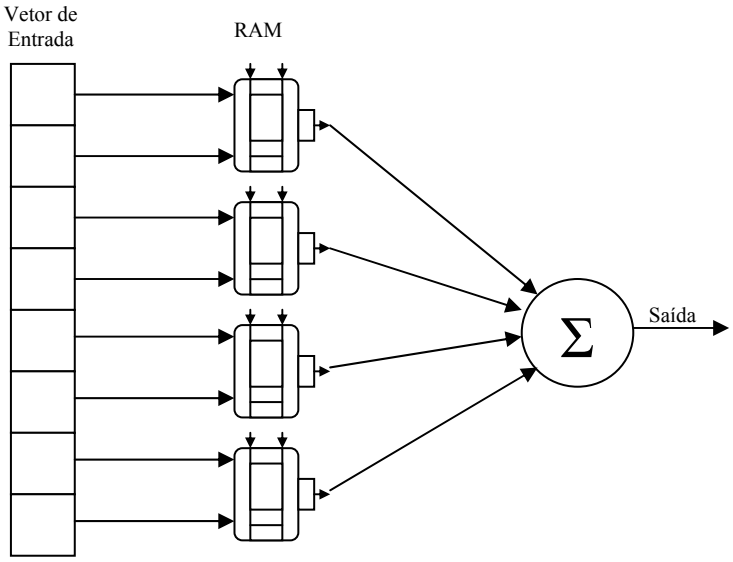


Figura 2.3 – Discriminador.

Após a fase de treinamento, em que todas as classes são treinadas apenas nos seus respectivos discriminadores, ocorre a fase de teste. Nesta fase, o reconhecimento de um padrão desconhecido *p'* é feito, apresentando o padrão *p'* a todos os discriminadores e verificando a resposta R_i de cada um dos discriminadores, sendo R_i a porcentagem de neurônios RAM do discriminador D_i que responderam com 1. O padrão *p'* será classificado como sendo da classe correspondente ao discriminador com a maior resposta.

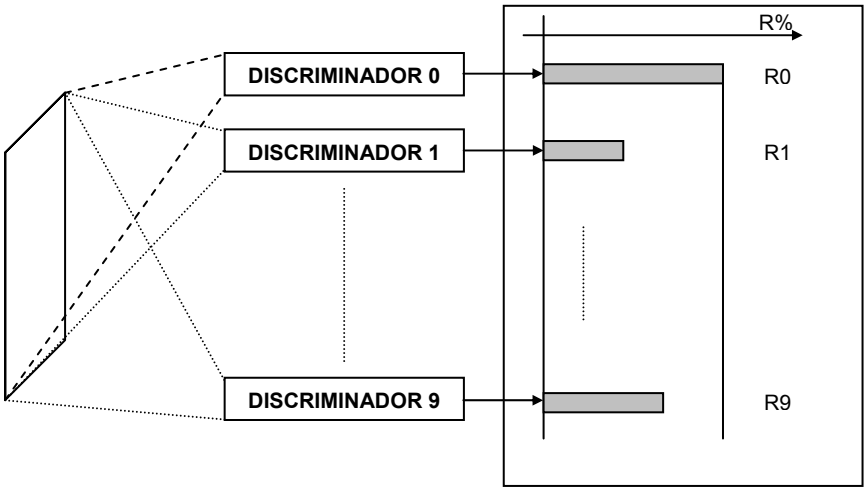


Figura 2.4 – Multidiscriminador.

Os principais parâmetros, que irão influenciar na performance dos multidiscriminadores, são a quantidade de conexões por neurônio e o padrão de conectividade. Em geral, uma quantidade de conexões grande é necessária, caso [STONHAM, 1984]:

1. os padrões de uma determinada classe forem muito similares;
2. os dados forem suscetíveis a ruído; ou
3. exista uma grande diversidade de padrões dentro de cada classe.

Em relação ao padrão de conectividade, pode ser obtido baseado no conhecimento de características específicas de uma determinada categoria de padrões. Como normalmente esse tipo de conhecimento não é disponível, geralmente a escolha das conexões é feita de forma aleatória para cada neurônio. [STONHAM, 1984]

Neste trabalho, como será visto no Capítulo 3, ao contrário do que é normalmente usado na literatura [BRAGA et al., 2000] [LUDERMIR et al., 1999], após o processo de otimização de um multidiscriminador, o número de neurônios RAM e o *fan-in* de cada neurônio RAM, em cada discriminador, podem variar.

2.3 Técnicas de Otimização Global

O objetivo desta seção é o de apresentar uma visão geral sobre os três métodos de otimização global usados neste trabalho: algoritmos genéticos [HOLLAND, 1992], *simulated annealing* [KIRKPATRICK et al., 1983] e *tabu search*. [GLOVER, 1986]

2.3.1 Algoritmos Genéticos

Técnicas de busca tradicionais iniciam seu processamento com uma única solução candidata que, iterativamente, é manipulada, usando alguma heurística, normalmente, relacionada ao problema a ser solucionado. Essas técnicas são utilizadas com sucesso, em várias aplicações. [WINSTON, 1992]

Entretanto, técnicas, como algoritmos genéticos, operam paralelamente com várias soluções candidatas, promovendo uma busca em diferentes regiões do espaço de solução ao mesmo tempo. Dessa forma, algoritmos genéticos têm maiores possibilidades de atingir áreas mais promissoras. As principais diferenças entre algoritmos genéticos e as técnicas de busca tradicionais são: não lidar com os parâmetros diretamente, mas sim, com uma codificação destes; trabalhar com uma população de soluções candidatas e não, com, apenas, uma solução; não necessitar de muito conhecimento do problema, apenas o suficiente para as funções de avaliação de soluções; gerar novas soluções de forma probabilística e não, determinística. [WINSTON, 1992]

A motivação inicial dos algoritmos genéticos é o modelo biológico de evolução. No início do processo, uma população de soluções possíveis é gerada aleatoriamente, formando a *população inicial*. Durante as iterações, chamadas de *gerações*, toda a população é avaliada e é associada uma nota a cada indivíduo, indicando o seu grau de adaptação como solução do problema. Os mais adaptados são selecionados para sofrerem modificações por alguns operadores genéticos, como mutação ou *crossover*, formando a próxima geração da população. [PHAM e KARABOGA, 2000] [JONG, 1980]

Em resumo, *algoritmos genéticos* (AG) são métodos de otimização global que empregam uma estratégia de busca paralela e estruturada, que, embora aleatória, direciona a soluções cada vez mais adaptadas. Por ter a capacidade de explorar vários pontos do espaço de busca simultaneamente, o risco de estacionamento em mínimos locais é reduzido. Esses métodos conseguem lidar bem quando o espaço de busca é amplo e complexo [PHAM e KARABOGA, 2000]. O fluxograma de um AG padrão é apresentado na Figura 2.5.

Para implementar o método para um determinado problema, existem quatro principais escolhas que devem ser feitas [Pham e Karaboga, 2000]: a representação das soluções; a definição da função de avaliação de aptidão; os operadores genéticos e a definição dos parâmetros de controle. Abaixo, é apresentado um resumo destes quatro assuntos.

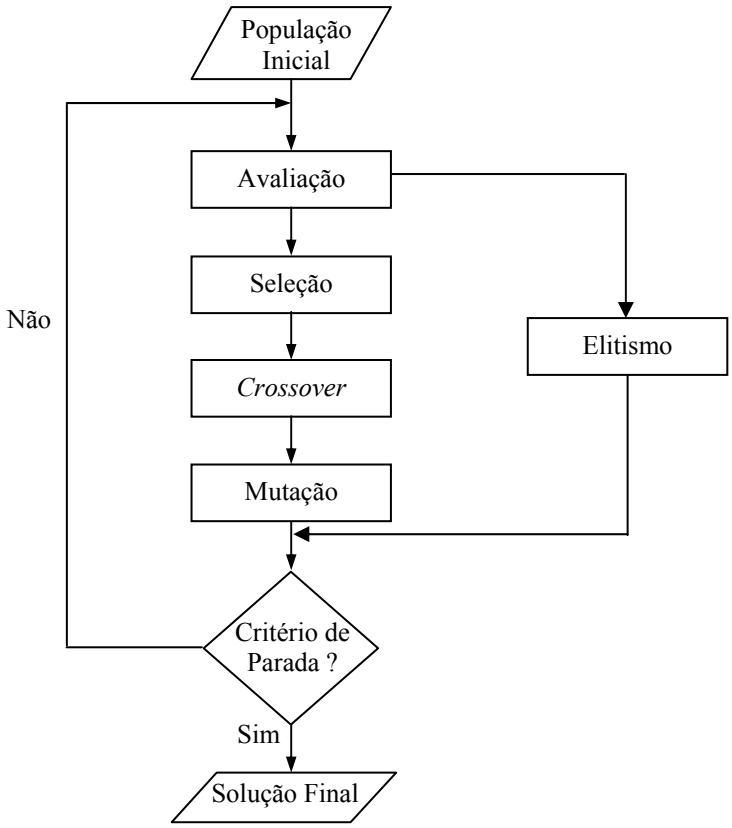


Figura 2.5 - Fluxograma de um AG simples.

Representação

A *representação* de uma solução, de forma que um AG possa trabalhar adequadamente, é um dos aspectos mais importantes do método. Essa representação provoca um grande impacto na performance, pois diversos esquemas de representação podem causar diferentes performances em termos da qualidade da solução e do tempo computacional. [PHAM e KARABOGA, 2000]

A representação de uma solução é chamada de *cromossomo* ou *indivíduo*. Existem dois métodos de representação bastante comuns na literatura [DAVIS, 1991] [MICHALEWICZ, 1992], sendo o mais comum o método de representação de cadeia binária. De um modo geral, o cromossomo representa o conjunto de parâmetros da função objetivo cuja resposta será otimizada. O conjunto de todas as configurações, que o cromossomo pode assumir, forma o espaço de busca do método. Se o cromossomo representa *n* parâmetros, o espaço de busca é *n*-dimensional. [CARVALHO et al., 2003]

Função de Avaliação de Aptidão

A *função de avaliação de aptidão* age como uma interface entre AG e o problema a ser otimizado, pois AG escolhe soluções de acordo com suas qualidades de acordo com a nota dada por esta função e não, por alguma informação, segundo as estruturas das soluções. Sendo assim, a qualidade de uma solução proposta é usualmente calculada, dependendo de como esta solução responde a funções desejadas e satisfaz restrições dadas. Dessa maneira, criar um procedimento automático que compute a qualidade das soluções é um outro aspecto bastante crucial do método. Por fim, a complexidade das funções de avaliação de aptidão dependerá muito do problema em si. [PHAM e KARABOGA, 2000]

Operadores Genéticos

Existem quatro tipos comuns de operadores genéticos: seleção, *crossover*, mutação e elitismo. A escolha ou a arquitetura do operador depende do problema e do esquema de representação empregado. [PHAM e KARABOGA, 2000]

Na *seleção*, o principal procedimento é escolher cópias de indivíduos cuja aptidão na resolução do problema é maior. Os indivíduos selecionados são armazenados em um conjunto intermediário da população, zerado a cada geração, conhecido como *mating pool*. O procedimento de seleção tem uma influência significativa em guiar a busca através de áreas com melhores resultados e encontrar boas soluções em um tempo curto. Contudo, a diversidade populacional deve ser mantida com o propósito de evitar a convergência prematura e alcançar a solução ótima global [PHAM e KARABOGA, 2000]. Os métodos de seleção mais usados são o *método da roleta* e o *método de* ***ranking***. [BRAGA et al., 2000]

No ***crossover*** (Figura 2.6), que é considerado o operador diferencial do método AG em relação aos outros métodos tradicionais de busca, dois novos indivíduos, chamados de *filhos*, são criados a partir de dois indivíduos existentes, chamados de *pais*, obtidos da população atual pelo operador de seleção. Os filhos substituem seus pais no *mating pool*. Vale salientar que esta operação não muda os valores dos bits [PHAM e KARABOGA, 2000]. Os tipos de *crossovers* mais freqüentes são o *crossover* de um ponto, o *crossover* multiponto e o *crossover* uniforme. [BRAGA et al., 2000]

Pai 1	**1 0 0 0 1 0 0 1 1 1 1**
Pai 2	0 1 1 0 1 1 0 0 0 1 1
Filho 1	**1 0 0 0 1** 1 0 0 0 1 1
Filho 2	0 1 1 0 1 **0 0 1 1 1 1**

Figura 2.6 - Exemplo de *crossover*.

Na *mutação*, todos os indivíduos da população do *mating pool*, após o *crossover*, são checados bit a bit. Essa checagem consiste em gerar um número aleatório, e, se esse número for menor que a taxa de mutação, o valor do bit será modificado. O operador de mutação força a procura de novas áreas no espaço de busca, assegurando que a probabilidade de se chegar a qualquer ponto do espaço de busca nunca será zero [BRAGA et al., 2000]. Eventualmente, ele ajuda o AG a evitar convergência prematura e achar uma solução ótima global. [PHAM e KARABOGA, 2000] (Figura 2.7)

Cadeia antiga	1 1 0 0 **0** 1 0 1 1 1 0
Cadeia nova	1 1 0 0 **1** 1 0 1 1 1 0

Figura 2.7 - Exemplo de mutação.

No *elitismo*, os *n* melhores indivíduos da geração atual substituem os *n* piores indivíduos da próxima geração. Este operador serve para guiar o espaço de busca, restringindo-o, convergindo o algoritmo mais rapidamente. Em contrapartida, diminui a diversidade da população, dificultando a investigação em certas regiões.

Parâmetros de Controle

O desempenho do método AG é fortemente influenciado por alguns parâmetros. Sendo assim, é de extrema importância analisá-los, de acordo com as necessidades do problema e dos recursos disponíveis [JONG, 1980]. Parâmetros de controle importantes de um simples AG incluem [PHAM e KARABOGA, 2000] [CARVALHO et al., 2003]:

1. *Tamanho da população* – uma grande população permite lidar simultaneamente com muitas soluções, embora aumente o tempo de computação e a quantidade de memória requerida por iteração. Contudo, ao usar muitas soluções ao mesmo tempo, a probabilidade de convergir para uma solução global ótima é maior do que com um tamanho pequeno da população.
2. *Taxa de **crossover*** – quanto maior a taxa, mais rapidamente novas estruturas são introduzidas na população. No início da otimização, uma alta taxa serve para descobrir

regiões promissoras, embora possa levar a uma saturação ao redor de uma solução. Uma baixa taxa, entretanto, diminui a velocidade de convergência para áreas promissoras.

3. *Taxa de mutação* – uma alta taxa introduz uma alta diversidade na população e pode causar instabilidade, tornando a busca essencialmente aleatória. Por outro lado, normalmente é muito difícil para um AG encontrar uma solução global ótima com uma taxa muito baixa.

2.3.2 *Simulated Annealing*

Enquanto os AGs são inspirados na evolução biológica, o método de *Simulated Annealing* [KIRKPATRICK et al., 1983] é inspirado na mecânica estatística, baseando-se na analogia entre os processos de resfriamento de sólidos (*annealing*) e os problemas de otimização. [PHAM e KARABOGA, 2000]

Annealing é o processo de resfriamento de cristais utilizado para alcançar energias mínimas, correspondentes a uma estrutura cristalina perfeita. Isto só é alcançado, se a estrutura for esfriada de forma suficientemente lenta. No início, os átomos dos cristais possuem alta temperatura, permitindo que ocorram diversas configurações. Com o passar do tempo, a temperatura vai diminuindo, e os cristais vão se acomodando, de modo a formar uma estrutura regular no formato do cristal, atingindo uma conformação de energia mínima. Caso a temperatura caia muito rapidamente, haverá irregularidades no cristal, pois o sistema não atingirá o estado de mínima energia. [PHAM e KARABOGA, 2000]

Pelo processo de *annealing*, a distribuição de probabilidade das energias do sistema em uma certa temperatura é determinada pela probabilidade de Boltzmann:

$$P(E) \propto e^{[-E/\ (kT)]} \tag{2.3}$$

em que E é a energia do sistema; k, a constante de Boltzmann; T, a temperatura; e $P(E)$, a probabilidade de que o sistema esteja em um estado com energia E. De acordo com a Equação 2.3, em temperaturas altas, $P(E)$ tende a 1 para todos os estados de energia. Também pode ser

notado que existe uma pequena probabilidade do sistema estar em um estado de energia alto, mesmo se a temperatura estiver baixa. Sendo assim, a distribuição de probabilidades permite que o sistema consiga escapar de mínimos locais de energia. [PHAM e KARABOGA, 2000]

Na analogia entre o processo de *annealing* e um problema de otimização, os estados do sólido representam as soluções possíveis para o problema, as energias dos estados correspondem aos custos das soluções, e o estado de mínima energia equivale a solução ótima para o problema. O método de ***Simulated Annealing*** (SA) funciona com uma seqüência de iterações, em que a solução atual é modificada aleatoriamente a cada iteração, de forma a criar uma nova solução possível, vizinha à solução atual. Tal vizinhança é determinada pelo mecanismo escolhido de geração de novas soluções.

Em seguida, é computado o valor da função de custo da nova solução, e a variação do custo é utilizada para decidir se a nova solução será aceita ou não. Se a mudança no custo for negativa, a nova solução é aceita; caso contrário, a nova solução pode ser aceita ou não, dependendo do *critério de Metrópolis* [METROPOLIS et al., 1953], baseado na Equação 2.3. De acordo com este critério, se a variação no custo for positiva ou nula, é gerado um número aleatório δ entre [0, 1], a partir de uma distribuição uniforme. Se $\delta \leq e^{(-\Delta E/\ T)}$, sendo ΔE a diferença de custo das duas soluções, então a nova solução é aceita como a nova solução atual; senão a solução original permanece.

O fluxograma de um algoritmo SA padrão é apresentado na Figura 2.8. Para implementar SA a um determinado problema, existem quatro principais escolhas que devem ser feitas [PHAM e KARABOGA, 2000]: a representação das soluções; a definição da função de custo; a definição do mecanismo de geração de vizinhos e a definição de um esquema de resfriamento. A representação da solução e a definição da função de custo são equivalentes, respectivamente, à codificação do cromossomo e à função de avaliação do método AG. Mecanismos de geração de uma nova solução também podem ser emprestados do método AG, como, por exemplo, o uso de mutação.

Em relação ao esquema de resfriamento, existem quatro parâmetros que devem ser especificados: temperatura inicial, regra de atualização de temperatura, número de iterações e o critério de parada. Diversos esquemas de esfriamento já foram propostos [OSMAN, 1991], sendo

amplamente utilizados esquemas que se utilizam de funções do tipo $T_{i+1} = cT_i$, i=0, 1..., sendo c um fator constante menor do que 1, embora, próximo a ele [PHAM e KARABOGA, 2000]. Quanto aos outros 3 parâmetros, temperatura inicial, número de iterações e o critério de parada, não há uma regra geral, variando de acordo com o problema.

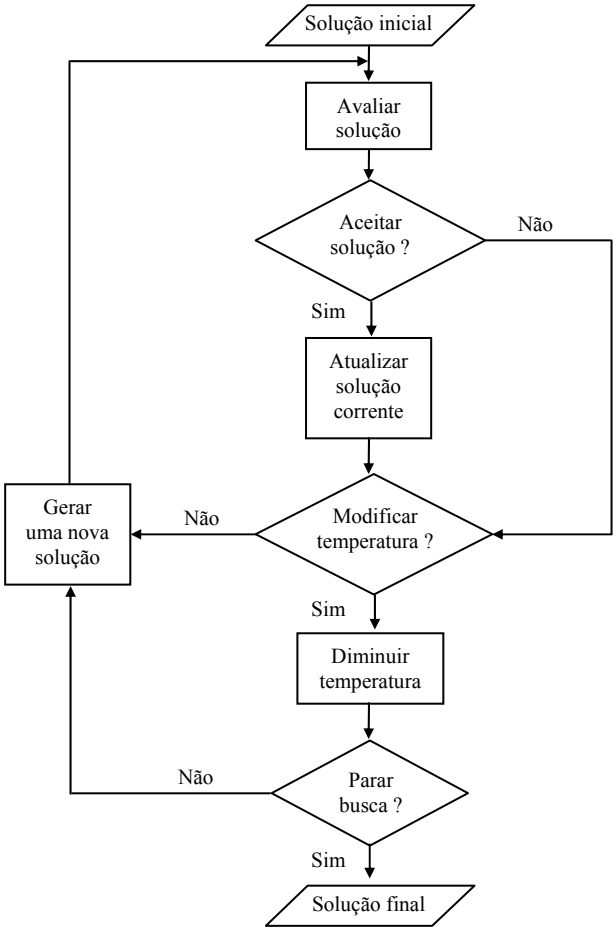


Figura 2.8 - Fluxograma de um algoritmo padrão de SA.

2.3.3 *Tabu Search*

O algoritmo ***Tabu Search*** (TS) foi desenvolvido independentemente por Glover [GLOVER, 1986] e Hansen [HANSEN, 1986] para resolver problemas de otimizações combinatoriais. O método TS é um tipo de algoritmo iterativo de busca, caracterizado pelo uso de uma memória flexível. Esse método também consegue eliminar mínimos locais e pesquisar áreas além de um

mínimo local. Dessa forma, o método TS tem a habilidade de descobrir o mínimo global de um espaço de busca multimodal. [PHAM e KARABOGA, 2000]

Nesse método, é escolhida a solução de maior avaliação dentro da vizinhança da solução atual que atenda às restrições tabus. Sendo assim, o método seleciona o movimento que produza a maior melhoria ou a menor deterioração na função de avaliação. Para isso, uma *lista tabu* é utilizada para armazenar os movimentos aceitos, permitindo classificar certos movimentos como proibidos em iterações posteriores. Dessa maneira, a lista tabu proíbe certas soluções de serem alcançadas a partir da solução atual [PHAM e KARABOGA, 2000].

Visto que os movimentos, que não melhoram a avaliação, podem ser aceitos, é possível que ocorra um retorno a soluções visitadas anteriormente. Isto pode causar eventuais ciclos que são evitados pelo uso da lista tabu. Uma estratégia de proibição é adotada para controlar e atualizar a lista tabu, descartando regiões já visitadas e forçando a exploração de novas regiões do espaço de busca. [PHAM e KARABOGA, 2000]

Idealmente, a lista tabu deveria guardar todas as soluções previamente visitadas e ser verificada antes de qualquer movimento. Contudo, essa abordagem requer muita memória e esforço computacional. Uma outra abordagem seria evitar, apenas, a última solução escolhida, sendo que ela é ineficaz, para garantir a não ocorrência de ciclos [PHAM e KARABOGA, 2000]. Então, na prática, a abordagem alternativa mais comum é a de evitar as T_s últimas soluções visitadas, sendo estas guardadas na lista tabu. O parâmetro T_s, conhecido como comprimento ou *tamanho da lista tabu*, deve ser cuidadosamente escolhido. Se for muito grande, a busca pode sair de regiões promissoras, antes de serem satisfatoriamente exploradas. Se for muito pequeno, a probabilidade de ocorrer ciclos aumenta [PHAM e KARABOGA, 2000]. Dessa maneira, a lista tabu armazena os T_s movimentos mais recentes. Quando a lista fica cheia, o novo movimento sobrepõe o movimento mais antigo, e a lista tabu acaba agindo como se fosse uma fila.

A lista tabu tem a função de restringir o espaço de busca do algoritmo. Normalmente, as proibições da lista são ativadas apenas quando o movimento ocorreu dentro de um número limitado de iterações anteriores à atual, denominada de *restrição baseada no tempo* ou quando o movimento ocorreu com uma certa freqüência sobre um dado número de iterações, denominada de *restrição baseada na freqüência*. [PHAM e KARABOGA, 2000]

Ao final de um certo número de iterações ou após a melhor solução ter sido encontrada, o algoritmo pára, e a solução, obtida por ele, será a melhor detectada até aquele momento [PHAM e KARABOGA, 2000]. Na Figura 2.9, é apresentado o fluxograma de um algoritmo TS padrão.

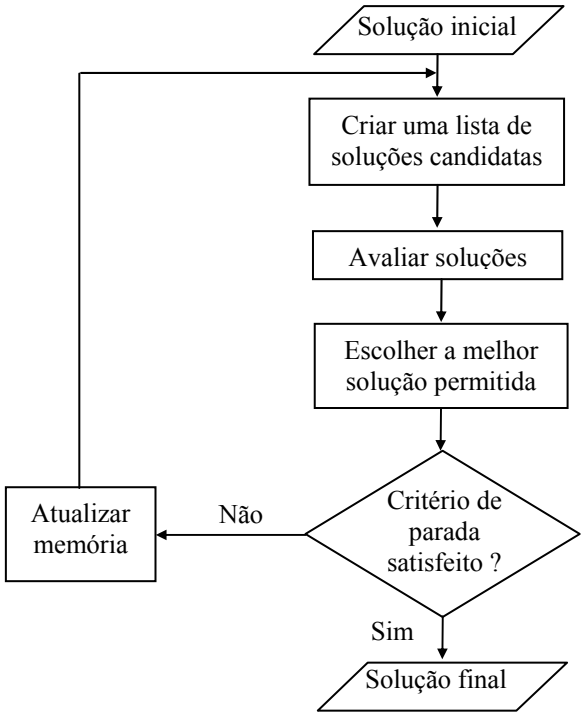


Figura 2.9 - Fluxograma de um algoritmo TS padrão.

Em comparação com SA, uma iteração de TS exige muito mais esforço computacional, embora, em geral, necessite de muito menos iterações para convergir, pois trabalha com um conjunto de soluções por iteração. [PHAM e KARABOGA, 2000]

2.4 Método de Treinamento Híbrido

O objetivo principal do trabalho é o de investigar a integração de duas técnicas, o algoritmo *one-shot-learning*, um método de otimização global no treinamento de multidiscriminadores. No contexto deste trabalho, o método de otimização global pode ser um dos três métodos de otimização, apresentados na Seção 2.3. O princípio geral desse treinamento, também descrito na Figura 2.10, baseia-se nas seguintes etapas:

1. **Solução inicial** – um multidiscriminador ou um conjunto de multidiscriminadores.
2. **Avaliação da solução** – a solução atual é treinada com o algoritmo *one-shot-learning* e avaliada por alguma função, sendo esse passo opcional dependente do método de otimização global.
3. **Geração de uma nova solução** – uma nova solução é gerada, a partir de pequenas modificações da solução atual. Essa nova solução pode ser um multidiscriminador ou um conjunto de multidiscriminadores, dependendo do método de otimização global.
4. **Avaliação da nova solução** – a nova solução é, então, treinada com o algoritmo *one-shot-learning* e avaliada por alguma função.
5. **Atualização da solução atual** – a nova solução poderá substituir a solução atual, de acordo com algum critério próprio de cada método de otimização global.
6. **Solução final** – se ocorrer o critério de parada, o melhor multidiscriminador avaliado será fornecido como saída.

Dessa forma, conclui-se que o método de treinamento híbrido pode ser visto da seguinte forma: a partir de uma solução inicial, o método de treinamento híbrido fornece como saída uma solução mais adaptada, de acordo com algum critério de avaliação.

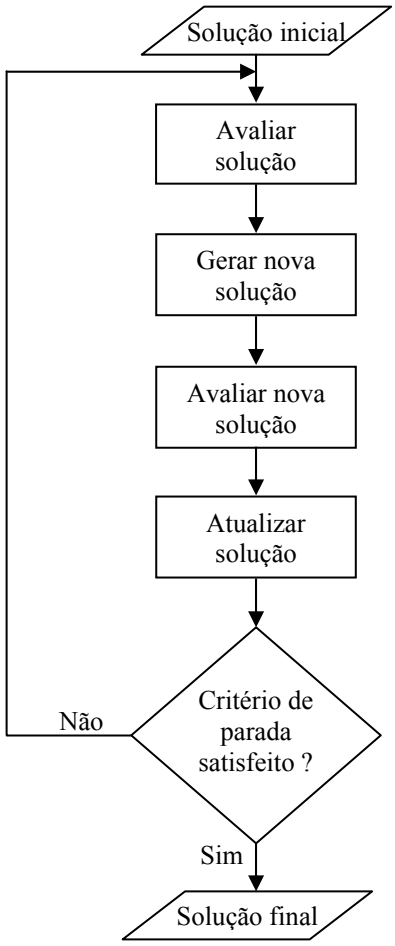


Figura 2.10 - Fluxograma geral da metodologia híbrida utilizada.

2.5 Trabalhos Relacionados

Diversos trabalhos na literatura vêm utilizando métodos de otimização global, tais como AG, SA e TS, para melhorar a performance de RNAs [BISHOP et al., 1990] [BRANKE, 1995] [CHALUP e MAIRE, 1999] [SEXTON et al., 1998] [YAMAZAKI et al., 2002] [YAO, 1999]. Vale salientar que a maior parte destes trabalhos utiliza AGs, embora isso não implique que essa técnica sempre gere bons resultados. Com relação aos outros dois métodos, também revisados nesta seção, SA e TS, apesar de existirem poucos trabalhos relacionados sobre o seu uso para a otimização de RNAs, os resultados obtidos têm sido bastante promissores. [BISHOP et al., 1990] [BRANKE, 1995] [YAMAZAKI et al., 2002]

A combinação de RNAs com AGs tem sido utilizada, principalmente, para otimizar a matriz de pesos, arquiteturas e parâmetros de aprendizado de RNAs [BRANKE, 1995] [PRADOS, 1992] [YAO, 1999]. Por exemplo, Prados [Prados, 1992] descreve o uso de AG, para treinar a matriz de pesos de forma mais rápida do que com o uso do *backpropagation*. Um outro exemplo [BAXTER, 1992], utilizou AG, para otimizar a matriz de pesos, a arquitetura e os parâmetros de aprendizado de uma RNA. Esse trabalho mostrou que o aprendizado de uma RNA poderia ser melhorado, utilizando o processo de evolução e que comportamentos complexos poderiam ser aprendidos.

Em [SEXTON et al., 1998], TS foi utilizado para otimizar a matriz de pesos de RNAs. Nesse experimento, sete tarefas diferentes foram utilizadas para serem compararados os resultados obtidos com o algoritmo de *backpropagation* e com o método TS. Os resultados obtidos indicaram que TS derivava soluções significantemente superiores e com convergência mais rápida. Além disso, os resultados obtidos com *backpropagation* apresentaram variâncias bem maiores do que os obtidos com TS, demonstrando que *backpropagation* ficara preso em mínimos locais subótimos.

Em [YAMAZAKI et al., 2002], SA foi utilizado para otimizar tanto a arquitetura quanto a matriz de pesos de uma MLP na tarefa de reconhecimento de odores, e obtendo, em média 1,68% de erro de classificação com 11,68 nodos. Em contraste, ao utilizar redes MLP com *backpropagation*, para classificar odores de três safras de vinho, a média dos resultados obtidos com 12 nodos na camada escondida foi de 36,28% na taxa de erro de classificação, e com 20 nodos foi de 15,74%. [YAMAZAKI, 2001]

No único trabalho encontrado na literatura no contexto das redes neurais sem peso [BISHOP et al., 1990], SA foi utilizado para otimizar o padrão de conectividade de redes de nodos RAM na tarefa de discriminação de caracteres similares. Um dos resultados obtidos no trabalho de Bishop verificou que a otimização, usando SA, melhorou em 50% na ortogonalidade dos discriminadores, obtendo, com isso, uma melhora de 10% na discriminação dos caracteres. Mesmo assim, a tarefa de reconhecimento utilizada não foi muito complexa: discriminar o caractere ‘e’ do ‘c’ e o caractere ‘i’ do ‘l’, usando uma única fonte impressa. Até o momento, não há, na literatura, nenhum trabalho que mencione o uso de AG ou TS com o modelo neural sem peso.

2.6 Considerações Finais

A maioria dos trabalhos na literatura utiliza modelos neurais com peso e algoritmos genéticos, para otimizar a matriz de pesos, arquiteturas e parâmetros de aprendizado [BAXTER, 1992] [BRANKE, 1995] [PRADOS, 1992] [YAO, 1999]. Mais recentemente, começaram a surgir alguns estudos, utilizando esses modelos neurais com o *Simulated Annealing* e o *Tabu Search*. [SEXTON et al., 1998] [YAMAZAKI et al., 2002]

Considerando o contexto dos modelos neurais sem peso, que é o foco desta dissertação, apenas um trabalho foi encontrado [BISHOP et al., 1990], em que se utilizou SA, para otimizar o padrão de conectividade de uma rede de nodos RAM, para resolver um *toy problem*.

Esta dissertação, por outro lado, propõe um estudo mais abrangente, em que três métodos serão usados, algoritmos genéticos, *simulated annealing* e *tabu search*, na tarefa de otimização do padrão de conectividades de redes neurais sem peso. Daí, o objetivo é o de investigar como tais métodos influem na performance dos multidiscriminadores em tarefas complexas, como o reconhecimento de caracteres numéricos manuscritos. Os resultados obtidos são comparados estatisticamente, indicando se as diferenças são significantemente superiores ou inferiores ou se são, apenas, diferenças acidentais.

Capítulo 3

3 Experimentos

O objetivo deste capítulo é o de apresentar e discutir os resultados obtidos pelos quatro métodos de treinamento – tradicional, AG, SA e TS – individualmente. Porém, antes de apresentar os resultados, serão descritos o problema de classificação abordado, os particionamentos dos padrões e o método estatístico usado para comparar amostras. Todos esses experimentos foram implementados em C#.Net©, e todos os dados foram armazenados em um servidor de banco de dados MS SQL Server 2000©. Os resultados foram, então, compilados em planilhas MS Excel© e as análises comparativas dos resultados obtidos entre os diferentes métodos são apresentadas no Capítulo 4.

3.1 Descrição do Problema e da Base de Dados

O problema proposto é o reconhecimento de caracteres numéricos manuscritos, extraídos de códigos postais americanos e britânicos. Cada caractere foi transformado em uma matriz binária de 16x24 pontos e armazenado como um vetor de 384 bits.

Os dados utilizados foram divididos em duas bases, sendo uma delas americana – denominada nesta dissertação de base de dados **Buffalo** – desenvolvida pelo Centro de Excelência para Análise e Reconhecimento de Documentos (CEDAR) da Universidade Estadual de Nova Iorque, em Búfalo [HULL, 1994] e a outra, britânica – denominada de base de dados **Essex** – compilada pela Universidade de Essex a partir de códigos postais fornecidos pelo *British Post Office* [CANUTO, 2001].

Cada base é subdividida em dez classes, uma para cada caractere numérico. Um resumo de como os exemplos estão divididos é mostrado na Tabela 3.1.

Classe	Quantidade de Exemplos	
	Buffalo	Essex
Número 0	1034	455
Número 1	945	832
Número 2	896	598
Número 3	860	574
Número 4	834	493
Número 5	793	503
Número 6	881	488
Número 7	841	461
Número 8	816	462
Número 9	811	476

Tabela 3.1 - Resumo dos exemplos.

3.1.1 Divisão do Conjunto de Dados

Como algumas classes possuem mais exemplos que outras, foi necessário gerar um novo agrupamento para cada base de dados, de modo que todas as classes tivessem a mesma quantidade de exemplos. Isto é feito, com vistas a evitar tendência para classes que possuam mais exemplos.

Sendo assim, para este trabalho, foram criados os agrupamentos:

- **B** para a base de dados Buffalo, contendo 793 exemplos distintos para cada classe e
- **E** para a base de dados Essex, contendo 455 exemplos distintos para cada classe.

Todos os exemplos pertencentes às classes dos novos agrupamentos (**B** e **E**) foram selecionados aleatoriamente das classes originais, cuidado esse que existe com o intuito de evitar qualquer tendência interna que exista nas classes originais.

Ao final, os agrupamentos **B** e **E** foram subdivididos em três partições fixas e sem sobreposição, **B0**, **B1** e **B2**, e **E0**, **E1** e **E2**. A partição **B0** é formada por 50% do agrupamento **B**, ou seja, 396 exemplos. As partições **B1** e **B2** são formadas por 25% do agrupamento **B**, com 198 e 199 exemplos, respectivamente. Em relação ao agrupamento **E**, a partição **E0** é formada por 50% do agrupamento, com 226 exemplos. Já as partições **E1** e **E2** são formadas por 25% do agrupamento **E**, com 113 e 116 exemplos, respectivamente.

Durante o processo de treinamento dos multidiscriminadores pelos métodos de otimização global, a cada iteração, a partição **B0** (ou **E0)** é usada pelo algoritmo *one-shot-learning* para treinar e, após o treinamento, a partição **B1** (ou **E1**) serve como parâmetro da função de avaliação dos multidiscriminadores. Nos multidiscriminadores gerados através do método tradicional ou nos multidiscriminadores resultantes das otimizações dos métodos de otimização global, as partições **B0** e **B1** (ou **E0** e **E1**) são usadas pelo algoritmo *one-shot-learning* para treinar e, após o treinamento, a partição **B2** (ou **E2**) serve para obter os resultados finais de erro de classificação.

3.2 Teste de Hipótese

Um dos objetivos deste trabalho é comparar os erros de classificação dos multidiscriminadores obtidos pelos métodos de otimização global e pelo método tradicional. Nessa comparação, serão indicados quais métodos obtêm multidiscriminadores com erros de classificação menores ou sem diferenças significativas. Contudo, normalmente um processo de comparação não é simples e claro devido ao julgamento subjetivo de quem compara. Isso se

deve à arbitrariedade do que são valores muito próximos e valores muito diferentes durante a comparação. Com o intuito de se obterem comparações menos ambíguas, essa subjetividade deve ser evitada.

Para exemplificar o problema de subjetividade, um dos resultados obtidos por este trabalho constituiu-se em dois conjuntos de valores de erros de classificação independentes, *X* e *Y*, com 30 valores por amostra. A amostra *X* possui uma média de 7,44% e um desvio padrão de 0,42%. A amostra *Y* possui uma média de 7,08% e um desvio padrão de 0,39%. Como se pode perceber, as duas amostras apresentaram médias de erros de classificação próximos. Para saber se essas duas amostras possuem uma diferença significante, sem basear em um julgamento subjetivo, deve-se utilizar um teste de hipótese estatístico.

Para isso, o teste de hipóteses estatístico é uma maneira formal de decidir se a diferença de desempenho dos resultados de duas amostras é significante ou acidental [DUDA et al., 2000], fornecendo um processo de comparação bem delineado com definições operacionais, o que evita subjetividade. Neste trabalho, o teste de hipótese, utilizado para comparar os resultados obtidos pelos experimentos, é o teste *t* de variância combinada.

3.2.1 Teste *t* de Variância Combinada

Esse teste estatístico é utilizado para comparar duas amostras (conjuntos de resultados), e, para melhor entendê-lo, é necessário explicar alguns conceitos estatísticos em que o teste se baseia. Entre eles, o teste utiliza o conceito de inferência estatística para obter estimativas sobre as características de uma população de resultados a partir de uma parcela dessa população (uma amostra). [LEVINE e BERENSON, 2000]

Considerando uma amostra *X* com *n* valores selecionados de uma população de valores, define-se a média aritmética da amostra como sendo $\overline{X} = \frac{\sum_{i=1}^{n} X_i}{n}$. $\overline{X}$ é uma estimativa da média aritmética populacional que normalmente é desconhecida e denominada de μ_X. Também define-

se a variância da amostra como sendo $S_X^2 = \frac{n\sum_{i=1}^{n} X_i^2 - \left(\sum_{i=1}^{n} X_i\right)^2}{n(n-1)}$. S_X^2 é uma estimativa da variância populacional σ_X^2. Nesse contexto, denomina-se de *nível de significância de erro da estimativa* (α) a probabilidade de erro que a estimativa amostral esteja incorreta. Por exemplo, α=5% significa que, se todas as amostras possíveis de um mesmo tamanho *n* fossem retiradas da população, 5% delas não iriam conter a verdadeira média aritmética da população no intervalo de confiança de (1- α). O *nível de confiança de acerto da estimativa* é, então, definido como o complemento do *nível de significância de erro da estimativa*, ou seja, (1- α). E define-se $z_{(1-\alpha)/2}$ como o valor correspondente a uma área de $(1-\alpha)/2$ desde o centro de uma distribuição normal padronizada (Figura 3.1). Por fim, define-se o *intervalo de confiança de* μ_X como sendo $\overline{X} \pm z_{(1-\alpha)/2}\frac{\sigma}{\sqrt{n}}$, ou então, $\overline{X} - z_{(1-\alpha)/2}\frac{\sigma}{\sqrt{n}} \le \mu_X \le \overline{X} + z_{(1-\alpha)/2}\frac{\sigma}{\sqrt{n}}$. [LEVINE e BERENSON, 2000]

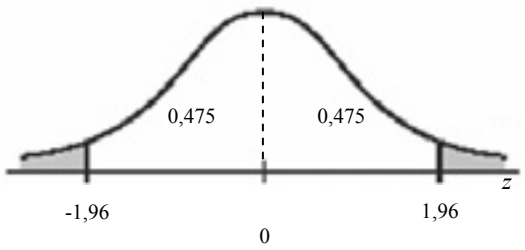


Figura 3.1 - Curva normal padrão, para determinar o valor de $z_{(1-\alpha)/2}$ necessário para o grau de confiança de (1- α)=95%, ou seja, $(1-\alpha)/2$ =0,475.

Todavia, o real desvio padrão da população σ normalmente é desconhecido. Assim sendo, uma forma de identificar o intervalo de confiança de μ_X deve ser obtido, usando somente as estatísticas de amostrais $\overline{X}$ e *S*. Inferir μ_X com σ desconhecido não utiliza a distribuição normal padrão, mas, uma distribuição conhecida como *t* de Student, que depende do tamanho amostral *n* [LEVINE e BERENSON, 2000]. Sendo assim, a *distribuição* ***t*** *de Student* dependerá do nível de confiança (1- α) e do *grau de liberdade*, definido como *n-1*. Quanto maior o tamanho da amostra, mais a distribuição *t* de Student se aproxima da distribuição normal padrão. Nesse

contexto, define-se o *intervalo de confiança de* μ_X, como sendo $\overline{X} \pm t_{(1-\alpha)/2,n-1} \frac{S}{\sqrt{n}}$, ou então [LEVINE e BERENSON, 2000],

$$\overline{X} - t_{(1-\alpha)/2,n-1} \frac{S}{\sqrt{n}} \leq \mu_X \leq \overline{X} + t_{(1-\alpha)/2,n-1} \frac{S}{\sqrt{n}} \quad \textbf{(3.1)}$$

reescrevendo,

$$-t_{(1-\alpha)/2,n-1} \leq \frac{\overline{X} - \mu_X}{\frac{S}{\sqrt{n}}} \leq t_{(1-\alpha)/2,n-1} \quad \textbf{(3.2)}$$

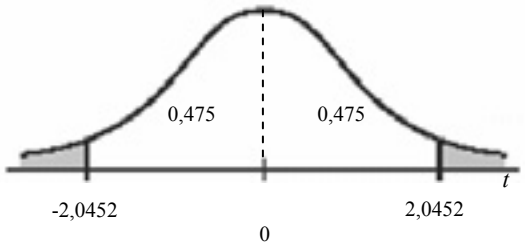


Figura 3.2 - Curva normal padrão, para determinar o valor de $t_{(1-\alpha)/2,n-1}$ necessário para o grau de liberdade *n*-1=29 e para o grau de confiança de (1-*α*)=95%, ou seja, $(1-\alpha)/2$ =0,475.

Todos esses conceitos acima descritos servem para inferir a média aritmética populacional μ, a partir de uma determinada amostra de valores com certo intervalo de confiança (1-α). Com o objetivo de comparar duas amostras independentes, esses conceitos básicos podem ser estendidos. A idéia básica é que, se as duas amostras possuem resultados próximos, a diferença de seus intervalos de confiança deve ser

$$-t_{(1-\alpha)/2,n_X+n_Y-1} \leq \frac{\overline{X} - \mu_X}{\frac{S_X}{\sqrt{n_X}}} - \frac{\overline{Y} - \mu_Y}{\frac{S_Y}{\sqrt{n_Y}}} \leq t_{(1-\alpha)/2,n_X+n_Y-1} \quad \textbf{(3.3)}$$

Como neste trabalho, todas as amostras possuem mesmo tamanho *n*, então $n_x = n_y = n$, e a Equação 3.3 pode ser simplificada para

$$-t_{(1-\alpha)/2,2n-1} \leq \frac{\left(\overline{X} - \overline{Y}\right) - \left(\mu_X - \mu_Y\right)}{\sqrt{\frac{S_X^2 - S_Y^2}{n}}} \leq t_{(1-\alpha)/2,2n-1} \quad \textbf{(3.4)}$$

Ao se compararem os dois intervalos de confiança, caso o módulo da diferença entre as duas mostras seja maior que $t_{(1-\alpha)/2,2n-1}$, pode-se chegar à conclusão de que eles são significantemente diferentes em nível de significância (α) escolhido. Entretanto, caso o módulo da diferença entre as duas mostras seja menor ou igual a $t_{(1-\alpha)/2,2n-1}$, não é possível afirmar relação de equivalência das amostras, mas, apenas, que não há diferenças significativas, detectadas em nível de significância (α) escolhido.

Esse teste que utiliza a distribuição *t* de Student, para comparar duas amostras independentes, é conhecido como teste *t* de variância combinada. Esse teste estatístico é robusto, pois não é sensível a distanciamentos moderados do pressuposto da normalidade das amostras [LEVINE e BERENSON, 2000]. Inicialmente, assume-se uma *hipótese nula*, denominada de H_0. Sempre que se determina a hipótese nula, deve-se especificar a hipótese alternativa, oposta à nula, denominada de H_1, que será verdadeira, caso H_0 seja falsa. [DUDA et al., 2000]

A hipótese nula pode ser a hipótese de que duas amostras não possuam diferenças significantes, denominado *teste bicaudal*, ou a hipótese de uma das amostras ser significantemente superior ou inferior que outra, denominado *teste unicaudal superior* ou *teste unicaudal inferior*, respectivamente. De acordo com um certo nível de significância (α), a distribuição de amostragem divide-se em duas regiões, uma de aceitação e outra de rejeição da hipótese nula (Figura 3.3). Caso o teste caia na região de aceitação, a hipótese nula não pode ser rejeitada. Caso contrário, ela é rejeitada, pois a região de rejeição consiste em valores que são improváveis de ocorrer, caso a hipótese nula fosse verdadeira. [LEVINE e BERENSON, 2000]

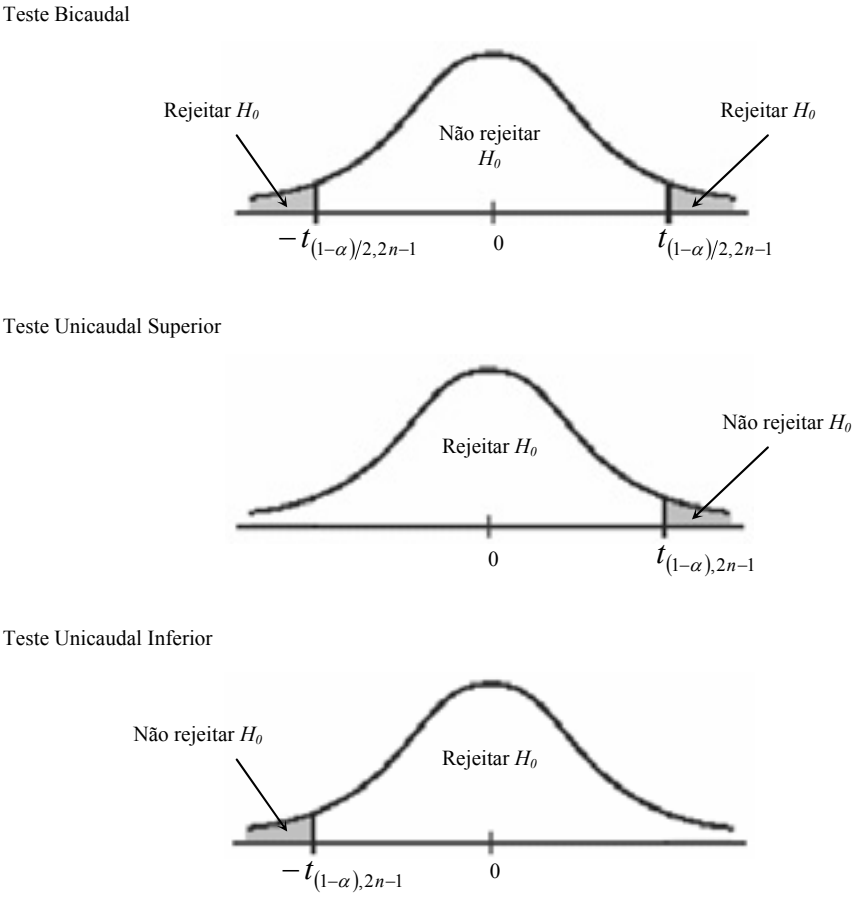


Figura 3.3 - Regiões de rejeição e não rejeição da hipótese nula para o teste de hipótese.

Teste Bicaudal

O objetivo do teste bicaudal é verificar se duas amostras possuem diferenças significativas. Nesse caso, a hipótese nula é que as médias aritméticas populacionais são iguais, e a hipótese alternativa é que elas são diferentes, ou seja, H_0: $\mu_X = \mu_Y$ ou $\mu_X - \mu_Y = 0$ e H_1: $\mu_X \neq \mu_Y$ ou $\mu_X - \mu_Y \neq 0$. Utiliza-se, então, a Equação 3.4 diretamente. Abaixo, será descrito como é feito o teste bicaudal.

Dados dois conjuntos, *X* e *Y*, de erros de classificação de tamanho *n*, são adotados os seguintes passos para o *teste bicaudal* [LEVINE e BERENSON, 2000]:

1. definir H_0: $\mu_X = \mu_Y$ e H_1: $\mu_X \neq \mu_Y$, em que μ_X e μ_Y são os valores médios reais de *X* e *Y*, respectivamente;
2. calcular os valores médios amostrais de *X* e *Y*, denominados de $\overline{X}$ e $\overline{Y}$;
3. calcular as variâncias amostrais de *X* e *Y*, denominadas de S_X^2 e S_Y^2;
4. calcular $t = \dfrac{\left(\overline{X} - \overline{Y}\right) - \left(\mu_X - \mu_Y\right)}{\sqrt{\dfrac{S_X^2 - S_Y^2}{n}}}$;
5. escolher um *nível de significância* (α), normalmente 1% ou 5%. O complemento do nível de significância (1- α) é conhecido como *nível de confiança* ou *intervalo de confiança* e é normalmente de 99% ou 95%; e
6. procurar na tabela *t-student* [LEVINE e BERENSON, 2000] o valor $t_{(1-\alpha)/2,2n-1}$, dado o *grau de liberdade*, 2*n*-1 e o nível de confiança $(1-\alpha)/2$. Caso $t > t_{(1-\alpha)/2,2n-1}$ ou $t < -t_{(1-\alpha)/2,2n-1}$, rejeita-se a hipótese nula, pois os valores são significantemente diferentes para o nível de confiança escolhido.

Teste Unicaudal

O objetivo do teste unicaudal superior é não só o de verificar se duas amostras possuem diferenças significativas, mas também indicar se a amostra *X* apresenta resultados significantemente maiores do que a amostra *Y*. Nesse caso, a hipótese nula é que a média aritmética populacional da amostra *X* é maior ou igual à da amostra *Y*, e a hipótese alternativa é que a média aritmética populacional da amostra *X* é menor que a da amostra *Y*, ou seja, H_0: $\mu_X \geq \mu_Y$ ou $\mu_X - \mu_Y \geq 0$ e H_1: $\mu_X < \mu_Y$ ou $\mu_X - \mu_Y < 0$. Para isso, é necessário alterar a Equação 3.4, desconsiderando o intervalo negativo e não mais, apenas, utilizar $(1-\alpha)/2$, mas,

sim, (1-α), pois, apenas, uma região é levada em consideração. Assim sendo, a Equação 3.4 é reescrita para

$$\frac{\left(\overline{X}-\overline{Y}\right)-\left(\mu_X-\mu_Y\right)}{\sqrt{\frac{S_X^2-S_Y^2}{n}}} \geq t_{(1-\alpha),2n-1} \qquad \textbf{(3.5)}$$

Dados dois conjuntos, X e Y, de erros de classificação de tamanho n, são adotados os seguintes passos para o *teste unicaudal superior* [LEVINE e BERENSON, 2000]:

1. definir H_0: $\mu_X \geq \mu_Y$ e H_1: $\mu_X < \mu_Y$, em que μ_X e μ_Y são os valores médios reais de X e Y, respectivamente;
2. repetir os passos de 2 a 5 do teste bicaudal; e
3. procurar na tabela *t-student* [Levine e Berenson, 2000] o valor $t_{(1-\alpha),2n-1}$, dado o *grau de liberdade*, 2n-1 e o nível de confiança (1-α). Caso $t < t_{(1-\alpha),2n-1}$, rejeita-se a hipótese nula, pois os valores da amostra X não são significantemente maiores do que os valores da amostra Y para o nível de confiança escolhido.

O objetivo do teste unicaudal inferior é não só verificar se duas amostras possuem diferenças significativas, mas também indicar se a amostra X apresenta resultados significantemente menores do que a amostra Y. Nesse caso, a hipótese nula é que a média aritmética populacional da amostra X é menor ou igual à da amostra Y, e a hipótese alternativa é que a média aritmética populacional da amostra X é maior que a da amostra Y, ou seja, H_0: $\mu_X \leq \mu_Y$ ou $\mu_X - \mu_Y \leq 0$ e H_1: $\mu_X > \mu_Y$ ou $\mu_X - \mu_Y > 0$. Para isso, é necessário alterar a Equação 3.4, desconsiderando o intervalo positivo e não mais, apenas, utilizar $(1-\alpha)/2$, mas, sim, (1-α), pois, apenas, uma região é levada em consideração. Assim sendo, a Equação 3.4 é reescrita para

$$\frac{\left(\overline{X}-\overline{Y}\right)-\left(\mu_X-\mu_Y\right)}{\sqrt{\frac{S_X^2-S_Y^2}{n}}} \leq -t_{(1-\alpha),2n-1} \qquad \textbf{(3.6)}$$

Dados dois conjuntos, X e Y, de erros de classificação de tamanho n, são adotados os seguintes passos para o *teste unicaudal inferior* [LEVINE e BERENSON, 2000]:

1. definir H_0: $\mu_X \leq \mu_Y$ e H_1: $\mu_X > \mu_Y$, em que μ_X e μ_Y são os valores médios reais de X e Y, respectivamente;
2. repetir os passos de 2 a 5 do teste bicaudal; e
3. procurar na tabela *t-student* [LEVINE e BERENSON, 2000] o valor $t_{(1-\alpha),2n-1}$, dado o *grau de liberdade*, 2n-1 e o nível de confiança (1-α). Caso $t > -t_{(1-\alpha),2n-1}$, rejeita-se a hipótese nula, pois os valores da amostra X não são significantemente menores do que os valores da amostra Y para o nível de confiança escolhido.

3.2.2 Pressupostos dos Experimentos

Alguns pressupostos são de extrema importância, para se poder fazer comparações entre os resultados obtidos pelos métodos [LEVINE e BERENSON, 2000]. Uma delas é não permitir que os resultados possuam qualquer tipo de viés ou qualquer tendência amostral sem correspondência populacional. Por isso, foram realizados os procedimentos para a geração dos agrupamentos descritos na Seção 3.1, pois, ao dar o mesmo número de padrões por classe e, ao escolher, aleatoriamente, os padrões da classe para a geração dos agrupamentos e partições, impediu-se que houvesse viés para alguma classe ou para alguma característica interna de uma classe.

Outro pressuposto é que todos os erros de classificação obtidos pelos multidiscriminadores tenham sido obtidos a partir dos mesmos conjuntos de entrada e saída. Por isso, esses erros de classificação identificados foram para uma mesma partição – **B2**, no caso dos multidiscriminadores Buffalo e **E2**, no caso dos multidiscriminadores Essex. Da mesma forma, o conjunto de treinamento para todos os multidiscriminadores também foi formado pelos mesmos conjuntos de partições, **B0** e **B1**, ou **E0** e **E1** para o caso dos multidiscriminadores Buffalo ou Essex, respectivamente. Todas essas partições são fixas e ordenadas.

Mais outro pressuposto importante é quanto à definição de uma política de descarte dos multidiscriminadores, obtidos pelos métodos de otimização global. Determinar uma política clara de descarte evita problemas éticos, como o de escolher, apenas, resultados que corroborem determinadas hipóteses. Neste trabalho, os multidiscriminadores, obtidos pelos métodos de otimização global, só eram descartados por um único motivo externo: quando o processamento

era cancelado no meio de alguma otimização de uma topologia, no caso de AG e de um multidiscriminador, no caso de SA e TS. Quando isso ocorria, as otimizações feitas pelo processamento eram invalidadas, refazendo-as de forma que todos os multidiscriminadores, obtidos pela otimização de cada topologia no caso de AG e pela otimização de cada multidiscriminador no caso de SA e TS, tivessem os mesmos parâmetros iniciais de otimização.

Vale salientar, também, que o nível de significância (α) escolhido foi de 5%, determinado no início do processo, antes da obtenção de qualquer resultado, pois não se deve permitir que, por motivos éticos, o nível de significância seja alterado após a coleta dos resultados para se obterem conclusões específicas. Esse valor de 5% foi escolhido para este trabalho, por ser um dos mais usados na literatura e o mais adequado para o problema em questão.

Todos os pressupostos acima são importantes para garantir que o teste de hipótese seja utilizado de maneira apropriada e que elimine a variabilidade decorrente a fatores externos, permitindo ter um maior nível de confiança nos resultados obtidos pelo teste de hipótese.

3.3 Experimentos com o Algoritmo Tradicional

3.3.1 Topologias

Neste trabalho, as topologias escolhidas, para serem utilizadas com o método tradicional, foram: topologias, iniciando com 16 neurônios, duplicando o número de neurônios até 128 (16, 32, 64 e 128) e topologias com 4 conexões por neurônio, aumentando de 4 em 4 até 16 conexões (4, 8, 12 e 16). Apenas a topologia de 128 neurônios com 16 conexões não foi gerada devido à falta de memória. Dessas quinze topologias diferentes, algumas delas possuem uma cobertura menor que o tamanho do vetor de entrada (384). Isso foi feito para apresentar uma uniformidade nos resultados tabelados e, assim, ter um maior embasamento para os comentários. A última topologia escolhida foi a de 300 neurônios com 6 conexões por neurônio, utilizada em um trabalho anterior [CANUTO, 2001]. Sendo assim, dezesseis topologias, obtidas pelo método tradicional, foram avaliadas para cada uma das bases de dados.

Com as topologias escolhidas, tendo por objetivo tirar conclusões estatísticas dos resultados, foram gerados 30 multidiscriminadores diferentes para cada topologia e base de dados (Buffalo e Essex). Os multidiscriminadores gerados foram armazenados em um banco de dados para posterior análise e uso pelos métodos de otimização global. As conexões dos discriminadores de cada multidiscriminador foram geradas aleatoriamente, sendo que a escolha dessas conexões obedeceu ao seguinte algoritmo:

1. Gerar o conjunto de conexões contendo {1, 2, ..., 384}.
2. Para cada neurônio:
    a. Remover uma nova conexão aleatoriamente do conjunto de conexões.
    b. Verificar se a nova conexão já existe no neurônio. Se existir, voltar ao Passo 2.a; se não existir, armazenar a nova conexão.
    c. Caso o conjunto de conexões estiver vazio, gerar novamente o conjunto contendo {1, 2, ..., 384}.
    d. Repetir o Passo 2.a, até que todas as conexões dos neurônios tenham sido escolhidas.

Esse mecanismo de geração de conexões impede que um mesmo neurônio se conecte a uma mesma conexão do vetor de entrada mais de uma vez. Isso garante que todas as conexões de um neurônio são diferentes.

3.3.2 Metodologia de Treinamento

O algoritmo *one-shot-learning*, utilizado para treinar os multidiscriminadores, está descrito na Seção 2.2.2. O conjunto de treinamento é formado pelas partições **B0** e **B1**, para os multidiscriminadores Buffalo, e **E0** e **E1**, para os Essex.

3.3.3 Aspecto Observado

A medida de erro utilizada para análise dos resultados foi o *erro de classificação* (EC), que corresponde à razão entre a quantidade de padrões classificados de forma incorreta e o número

total de padrões. Assim, o aspecto observado, ao final do treinamento, foi a percentagem do EC dos conjuntos **B3** e **E3**, respectivamente, para os multidiscriminadores Buffalo e Essex.

3.3.4 Resultados Obtidos

Para cada base de dados, é apresentada uma tabela resumida, contendo a média, o desvio padrão e o melhor resultado do aspecto observado nos 30 multidiscriminadores de cada topologia. Comentários do desempenho são feitos logo em seguida.

BUFFALO

Conexões	16 Neurônios			32 Neurônios			64 Neurônios			128 Neurônios		
	Média	Desvio	Melhor	Média	Desvio	Melhor	Média	Desvio	Melhor	Média	Desvio	Melhor
4	99,75	0,28	98,94	98,92	0,55	97,73	93,79	1,43	90,71	83,91	2,44	79,55
8	51,22	3,65	44,34	33,75	2,18	29,70	22,05	1,56	18,69	15,44	1,26	13,28
12	20,31	2,24	15,76	11,99	1,17	8,94	8,45	0,59	7,27	6,85	0,38	6,11
16	15,41	1,11	13,23	9,85	0,63	8,74	7,52	0,40	6,82	-	-	-

(a)

Conexões	300 Neurônios		
	Média	Desvio	Melhor
6	24,95	1,21	22,78

(b)

Tabela 3.2 – Erro de classificação, base de dados Buffalo (%).

Apenas duas topologias – 128 neurônios com 8 conexões em comparação a 16 neurônios, com 16 conexões – mostradas na Tabela 3.2, apresentaram diferenças significativas nos erros médios de classificação, com um nível de significância de 5%. De um modo geral, os desvios padrões em relação às médias foram pequenos, indicando uma certa homogeneidade das topologias.

Vale salientar que, mantendo constante o *fan-in* e aumentando a quantidade de neurônios, há uma diminuição significativa, com um nível de significância de 5%, no EC médio. Isso acontece, porque, com o incremento de nodos, há uma maior probabilidade de partes relevantes dos padrões de treinamento das classes serem aprendidas pelos nodos, levando em consideração que o padrão de conectividade é gerado uniformemente. Essa diminuição no EC médio, também, acontece em relação ao melhor resultado.

Esse comportamento também ocorre, mantendo-se constante a quantidade de neurônios e aumentando o *fan-in*. Nesse caso, o aumento do *fan-in* permite que mais características possam ser discriminadas por cada neurônio. Sem contar que, como se percebe na Tabela 3.2, o número de padrões no conjunto de treinamento (594 padrões por classe, formado pelas partições **B0**, **B1** e **B2**), foi suficientemente grande, para impedir uma superdiscriminação dos multidiscriminadores a esse conjunto com o aumento de *fan-in*. Além disso, um *fan-in* pequeno aumenta a chance das memórias dos neurônios saturarem.

ESSEX

Conexões	16 Neurônios			32 Neurônios			64 Neurônios			128 Neurônios		
	Média	Desvio	Melhor	Média	Desvio	Melhor	Média	Desvio	Melhor	Média	Desvio	Melhor
4	96,91	0,98	94,42	91,94	1,52	89,20	80,78	2,13	75,40	63,45	2,66	59,82
8	32,50	2,42	28,32	20,40	1,43	17,26	14,57	1,04	12,57	11,30	0,59	9,91
12	19,06	1,03	17,52	13,06	0,72	11,95	9,84	0,72	8,50	8,45	0,45	7,26
16	21,88	1,31	19,03	15,41	1,02	13,27	11,58	0,54	10,18	-	-	-

(a)

Conexões	300 Neurônios		
	Média	Desvio	Melhor
6	15,25	1,00	13,63

(b)

Tabela 3.3 – Erro de classificação, base de dados Essex (%).

Na Tabela 3.3, apenas quatro topologias não apresentaram diferenças significativas nos erros médios de classificação – 32 neurônios com 16 conexões em comparação a 300 neurônios

com 6 conexões e 64 neurônios com 16 conexões em comparação a 128 neurônios com 8 conexões –, com um nível de significância de 5%. De um modo geral, os desvios padrões em relação às médias também indicaram uma homogeneidade dos erros de classificação obtidos.

Também, ao se manter constante o *fan-in* e aumentar a quantidade de neurônios, há uma diminuição significativa, com um nível de significância de 5%, no erro médio de classificação. A explicação desse comportamento é análoga ao que ocorreu com a base Buffalo. Essa diminuição também acontece em relação ao melhor EC e ao desvio padrão. Contudo, mantendo-se constante a quantidade de neurônios e aumentando o *fan-in*, os valores (média, desvio padrão e melhor resultado dos erros de classificação) diminuem até 12 conexões por neurônio; ao se passar de 12 para 16 conexões por neurônio, há um aumento significativo dos erros de classificação.

Esses aumentos nos erros de classificação ocorreram devido ao incremento no *fan-in*, o que ocasionou uma superespecialização dos neurônios. Para que ocorresse uma generalização apropriada, era necessária uma maior diversidade de padrões no conjunto de treinamento. Como o conjunto de treinamento da base Essex é de 339 exemplos para cada classe – no conjunto formado pelas partições **E0**, **E1** e **E2** –, os discriminadores não tiveram um número suficiente de padrões para generalizar. Diferentemente, os discriminadores da base Buffalo, treinados com as partições **B0**, **B1** e **B2** – 594 exemplos para cada classe –, conseguiram diminuir os erros de classificação ao aumentar o *fan-in* de 12 para 16, pois a maior quantidade de exemplos, no conjunto de treinamento, possibilitou um melhor aproveitamento das memórias dos neurônios.

3.3.5 Topologias Escolhidas para Otimização

Para se avaliarem os métodos de otimização global AG, SA e TS, foram escolhidas três topologias, para serem otimizados seus padrões de conectividades, com o objetivo de se investigar se existem diferenças significativas na taxa de erro de classificação. As três topologias escolhidas foram as seguintes:

1. **64 neurônios com 16 conexões** – essa topologia foi escolhida por possuir o maior número de conexões por neurônio. Como os métodos de otimização global podem

desativar conexões e neurônios, modificando a arquitetura inicial, daqui por diante esta topologia será denominada de **T0.**

2. **128 neurônios com 12 conexões** –topologia escolhida por ter obtido os melhores resultados. Pelo mesmo motivo, observado no Item 1, daqui por diante, esta topologia será denominada de **T1.**
3. **300 neurônios com 6 conexões** – a opção por essa topologia se deve ao fato de possuir mais neurônios e a maior cobertura, 1800 bits. Isso permite fazer uma análise da performance dos métodos de otimização em relação a essas topologias. Pelo mesmo motivo observado no Item 1, daqui por diante essa topologia será denominada de **T2.**

3.4 Experimentos com Algoritmos Genéticos

Antes de apresentar os resultados obtidos, usando AG, é importante descrever os procedimentos usados e os elementos principais, conforme descrito na Seção 2.3.1, que são a representação das soluções, a função de avaliação de aptidão, os operadores genéticos e os parâmetros de controle.

3.4.1 Elementos Principais do Algoritmo

Representação

A representação escolhida neste trabalho, para codificar um neurônio RAM, é formada por dois vetores de mesmo tamanho. O primeiro é um vetor binário, indicando o status da conexão, se ativado ou desativado, denominado de *status*. O segundo é um vetor inteiro que contém valores de 1 a 384, indicando a qual índice do vetor de entrada se refere a conexão, denominado de *conexão*. Essa representação permite, mesmo com o *status* desativado, saber a qual índice do vetor de entrada o neurônio estava previamente conectado e voltar a ativá-lo, caso seja motivo de interesse.

O conjunto de neurônios, codificados de um discriminador, é a representação de um discriminador. O conjunto dos discriminadores codificados de um multidiscriminador forma a representação de um multidiscriminador.

Função de Avaliação de Aptidão

A função de avaliação de aptidão dos indivíduos é o erro de classificação para as partições **B2** e **E2**[1], respectivamente, para os multidiscriminadores Buffalo e Essex. Vale salientar que, antes de definir a função de avaliação de aptidão descrita acima, várias funções foram testadas. Essas funções eram somas ponderadas entre o erro de classificação e uma função que calculava o custo da matriz de conectividades. Diversas funções para calcular o custo da matriz de conectividades foram usadas. Diversos pesos, na soma ponderada, foram empregados para os valores de erro de classificação e para a função que calculava o custo da matriz, sendo que os melhores resultados eram obtidos, quando a função de avaliação de aptidão levava, em consideração, apenas, o erro de classificação.

Operadores Genéticos

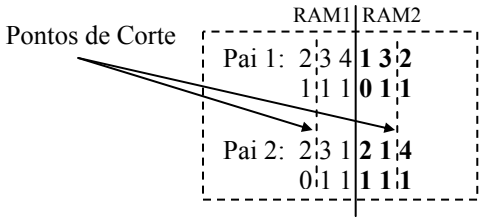
O método de seleção utilizado neste trabalho é o *método da roleta* [BRAGA et al., 2000]. Nele, cada indivíduo ocupa uma área proporcional à sua aptidão em uma roleta. Indivíduos mais aptos possuem maior probabilidade de serem selecionados. São selecionados 30 indivíduos em cada geração.

O método de *crossover*, utilizado neste trabalho, é descrito da forma a seguir: a partir de duas codificações (genótipos) pais, para cada *i*-ésimo genótipo de neurônio de cada discriminador de cada pai, gera-se o ponto de corte, aleatoriamente, de zero ao número de conexões do neurônio. Como neste trabalho não se permitiu um neurônio estar conectado para uma mesma conexão do vetor de entrada mais de uma vez, após o *crossover* de um ponto, caso

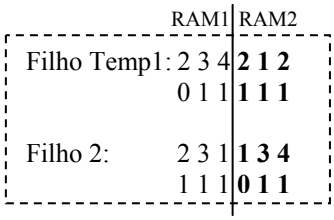
[1] No contexto da função de avaliação de aptidão, e, apenas, nele, o erro de classificação sempre será em relação às partições **B2** e **E2**, respectivamente, para os multidiscriminadores Buffalo e Essex.

algum neurônio filho possua mais de uma conexão com uma mesma entrada ativada, apenas, a primeira conexão dessa entrada permanece ativa, sendo as outras desativadas.

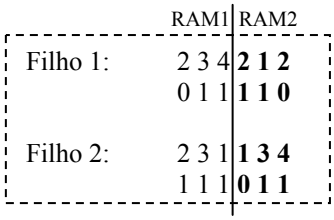
Na Figura 3.4, é apresentado um exemplo de *crossover*, sendo duas redes pais de dois neurônios cada, com um *fan-in* igual a três. Vale salientar que o último genótipo de neurônio do Filho 1 teve uma conexão desativada, por já possuir uma conexão igual ativa no mesmo neurônio (RAM2 do Filho 1).



(a) Posição dos pontos de corte.



(b) Filhos após o *crossover*.



(c) Filhos após desativação de conexões iguais.

Figura 3.4 - Exemplo de *crossover*.

O método de mutação, utilizado neste trabalho, é descrito da forma a seguir: para cada par do genótipo (*status* x *conexão*), gera-se um número aleatório. Caso esse número seja menor que a *taxa de mutação*, o par sofrerá mutação. Na mutação, se o *status* estiver ativado, ele será

desativado; se o status estiver desativado, ele será ativado, sendo gerado, aleatoriamente, um novo valor para a conexão. Isso permite que novos padrões de conectividades sejam testados.

Os valores possíveis para a taxa de mutação e a taxa de *crossover* serão descritos, a seguir, nos Parâmetros de Controle.

Figura 3.5 - Exemplo de mutação.

Parâmetros de Controle

Neste trabalho, o tamanho da população é constante e igual a trinta. A definição do conjunto dos melhores valores para a taxa de *crossover* foi obtida depois de vários testes, com taxas, variando em (0,999; 0,99; 0,95; 0,9; 0,85; 0,8; 0,75; 0,7; 0,6; 0,5; 0,25; 0,1; 0). O conjunto (0,95; 0,85; 0,75; 0,6) foi escolhido como o melhor para a taxa de *crossover* e denominado de *conjunto crossover*. Já o conjunto dos melhores valores para a taxa de mutação foi obtido depois de vários testes, com taxas, variando em (0,9; 0,5; 0,4; 0,3; 0,2; 0,15; 0,1; 0,05; 0,01; 0,008; 0,005; 0,003; 0,001; 0). O conjunto (0,01; 0,005; 0,001) foi escolhido como o melhor para a taxa de mutação e denominado de *conjunto mutação*.

3.4.2 Metodologia de Treinamento

A população inicial do AG é formada pelos 30 multidiscriminadores, previamente armazenados, gerados pelo método tradicional descrito na Seção 3.3.1. A cada nova geração, o treinamento de cada indivíduo da população é feito, usando as partições **B0** e **B1** para os multidiscriminadores Buffalo e **E0** e **E1**, para os multidiscriminadores Essex – utilizando o algoritmo *one-shot-learning*, descrito na Seção 2.2.2.

Após o treinamento, os indivíduos mais adaptados, de acordo com a função de avaliação de aptidão, são selecionados, aplicados o *crossover* e a mutação. Sendo assim, são obtidos os indivíduos da nova geração. Também foi observado que bons indivíduos eram descartados muito prematuramente. Para resolver esse problema, empregou-se elitismo a fim de se obterem erros de classificação menores. Foram feitos testes, sempre passando o indivíduo mais adaptado, os dois indivíduos mais adaptados e os três indivíduos mais adaptados da geração *t* para a geração *t+1*. Os menores erros de classificação foram obtidos, ao se substituir o pior indivíduo da próxima geração pelo melhor da geração atual.

Para determinar o par de valores *taxa de crossover* x *taxa de mutação*, foi adotada para cada topologia uma execução prévia do algoritmo com apenas 50 gerações, em que a taxa de *crossover* variava entre os valores do conjunto *crossover*, e a taxa de mutação variava entre os valores do conjunto mutação. Para cada topologia, o par *taxa de crossover* x *taxa de mutação*, que obtinha o menor erro de classificação, era o escolhido para otimizar a topologia.

	Buffalo		Essex	
	Mutação	*Crossover*	Mutação	*Crossover*
T0	0,1	95	0,5	60
T1	0,1	60	0,1	60
T2	0,5	75	1,0	85

Tabela 3.4 - Taxas de mutação e de *crossover* para cada topologia e base de dados selecionadas pela execução prévia (%)

A quantidade de gerações do algoritmo foi determinada após alguns testes, com gerações variando entre os valores (10, 30, 50, 100, 150, 200, 250, 300, 350, 400, 450, 500, 1000, 1500, 2000, 2500, 3500, 6000). Observou-se que os melhores resultados foram obtidos com 200 gerações. Como não se observou homogeneidade na população, após as 200 gerações e devido a um maior custo computacional, ao incluir essa verificação a cada geração, definiu-se o número de geração como único critério de parada.

Levando em consideração todos os procedimentos descritos acima, após 200 gerações, o algoritmo pára e, então, é verificado qual é o indivíduo mais adaptado da população (aquele que apresentar o menor valor obtido pela função de avaliação de aptidão). Nesse momento, esse

indivíduo tem sua memória limpa e, então, é treinado mais uma vez pelo algoritmo *one-shot-learning*, descrito na Seção 2.2.2, com o conjunto de treinamento, formado pelas partições **B0**, **B1** e **B2**, para os multidiscriminadores Buffalo, e **E0**, **E1** e **E2**, para os Essex.

Por fim, vale salientar que, durante o processo de otimização, para cada geração, o algoritmo de treinamento dos multidiscriminadores é um algoritmo híbrido, dividido em duas fases, formadas pelo algoritmo *one-shot-learning* e pela função de avaliação de aptidão do método AG. Mais precisamente, na primeira fase, é utilizado o algoritmo *one-shot-learning*, que usa como conjunto de treinamento as partições **B0** e **B1** para os multidiscriminadores Buffalo, e as partições **E0** e **E1**, para os Essex. Nessa fase, os multidiscriminadores decoram os padrões dessas partições. Já na segunda fase, a função de avaliação de aptidão direciona o espaço de busca. Os multidiscriminadores que possuem um padrão de conectividade mais adaptado a reconhecer os padrões das partições **B2** e **E2** (menor valor obtido pela função de avaliação de aptidão), respectivamente, para os multidiscriminadores Buffalo e Essex, terão maior probabilidade de serem selecionados para a próxima geração. Sendo assim, esse algoritmo apresenta uma tendência a *overfitting* para as partições **B2** e **E2**, respectivamente, para os multidiscriminadores Buffalo e Essex.

3.4.3 Aspecto Observado

É o mesmo aspecto descrito na Seção 3.3.3.

3.4.4 Resultados Obtidos

Para cada base de dados, foram feitas 30 execuções do método AG para cada topologia, tendo como população inicial os 30 multidiscriminadores, gerados na Seção 3.3.1. Essas 30 execuções tinham um mesmo par de valores *taxa de crossover* x *taxa de mutação* e geraram 30 novos multidiscriminadores para cada uma das três topologias escolhidas e das bases de dados. Abaixo, são apresentadas duas tabelas para cada base de dados, contendo a média, o desvio

padrão e o melhor resultado do aspecto observado e do *fan-in* médio nesses multidiscriminadores resultantes. Comentários do desempenho são feitos logo em seguida.

BUFFALO

	Topologias		
	T0	**T1**	**T2**
Média	7,44	7,08	23,99
Desvio Padrão	0,42	0,39	0,79
Melhor Resultado	6,41	6,31	22,68

Tabela 3.5 – Erro de classificação das 3 topologias otimizadas pelo método AG, base Buffalo (%).

	Topologias		
	T0	**T1**	**T2**
Média	14,66	11,33	5,31
Desvio Padrão	0,58	0,29	0,13
Melhor Resultado	13,55	10,65	4,97

Tabela 3.6 – *Fan-in* médio das 3 topologias otimizadas pelo método AG, base Buffalo.

Da Tabela 3.5, observa-se que quanto menor a média, menor o desvio padrão. O desvio padrão foi baixo, indicando uma uniformidade dos resultados obtidos. De um modo geral, a diferença entre a média e o melhor resultado ficou em torno de 1%. Todas as três topologias apresentaram diferenças significativas entre as médias dos erros de classificação, para um nível de significância de 5%. Da Tabela 3.6, percebe-se que a topologia **T0** teve seu *fan-in* diminuído de 16 para 14,66 em média, indicando que a topologia estava superdimensionada.

ESSEX

	Topologias		
	T0	**T1**	**T2**
Média	11,37	8,71	13,83
Desvio Padrão	1,09	0,57	0,92
Melhor Resultado	9,73	7,26	12,48

Tabela 3.7 – Erro de classificação das 3 topologias otimizadas pelo método AG, base Essex (%).

	Topologias		
	T0	**T1**	**T2**
Média	12,89	11,35	5,27
Desvio Padrão	0,74	0,39	0,16
Melhor Resultado	11,17	10,73	4,90

Tabela 3.8 – *Fan-in* médio das 3 topologias otimizadas pelo método AG, base Essex.

Todas as três topologias apresentaram diferenças significativas entre as médias dos erros de classificação, para um nível de significância de 5%. Da Tabela 3.7, observa-se que o desvio padrão das topologias otimizadas não manteve a tendência de que quanto menor a média, menor o desvio padrão. A razão para esse aumento do desvio padrão do EC da topologia **T0** é a diminuição do *fan-in*, conforme indicado na Tabela 3.8, passando de 16 para 12,89. Para a topologia **T0**, essa variação de *fan-in* levou a um aumento do espaço de busca (multidiscriminadores com *fan-in* médio de 11 a 16 conexões foram testados), havendo, assim, uma maior variedade nos erros de classificação das soluções geradas pelo método AG.

3.5 Experimentos com *Simulated Annealing*

Antes de apresentar os resultados obtidos, usando SA, é importante descrever os procedimentos usados e os elementos principais, conforme descrito na Seção 2.3.2, que são a representação das soluções, a função de custo, o método de geração de novos vizinhos e os parâmetros de controle.

3.5.1 Elementos Principais do Algoritmo

Representação e Função de Custo

A representação das soluções e a função de custo para o método SA são, respectivamente, a representação e a função de avaliação de aptidão usadas para o método AG, descritas na Seção 3.4.1.

Geração de Novos Vizinhos

Neste trabalho, um novo vizinho é gerado da forma a seguir (Figura 3.6): para cada par da codificação (*status* x *conexão*), ocorre um teste a fim de verificar se haverá ou não uma mudança, de acordo com a *taxa de mudança da solução atual*; caso ocorra, gera-se, aleatoriamente, um valor de zero ao tamanho do vetor de entrada. Se for zero, o status da conexão é desativado; se for diferente de zero, ativa-se o *status* e atualiza-se a conexão para o valor gerado, permitindo que novos padrões de conectividade sejam testados.

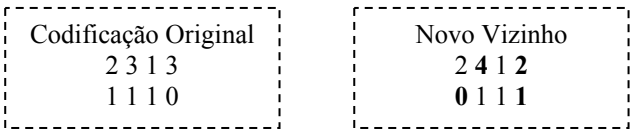


Figura 3.6 - Exemplo de geração de um novo vizinho.

A principal diferença entre o procedimento de gerar novos vizinhos do método SA e o operador de mutação do método AG, descrito na Seção 3.4.1, é que o procedimento do método SA tem uma menor probabilidade de desativar conexões, enquanto que, no método AG, no momento da mutação, o *status* é desativado, caso ele esteja ativo. No método SA, no momento em que ocorre a mudança no par (*status* x *conexão*), o *status* só será desativado, caso o resultado da geração de um número aleatório (variando de zero ao tamanho do vetor de entrada) seja zero.

A decisão de usar esse procedimento foi tomada, porque, nos primeiros testes, o método SA não conseguia convergir, ao utilizar o procedimento de mutação, descrito na Seção 3.4.1, para gerar novos vizinhos. A dificuldade em convergir deve-se ao fato de, apenas, um novo vizinho ser gerado a cada iteração. Esse novo vizinho tendia a ter a maioria de suas conexões desativadas nas primeiras iterações, ocasionando problemas de saturação nas memórias dos neurônios (a maioria das posições de memória ocupadas com 1). E, mesmo que possuíssem uma função de custo maior do que a solução atual, esses novos vizinhos eram aceitos com alta probabilidade nas primeiras iterações. No caso do método AG, esse problema era evitado, pois havia um conjunto de soluções a cada geração, e as soluções mais adaptadas tinham maior probabilidade de transferir suas características para as novas soluções na geração posterior.

Parâmetros de Controle

O *esquema de resfriamento* é uma função do tipo $T_{i+1} = cT_i$, sendo *c* uma constante durante toda a iteração. Testes foram feitos para se determinarem valores para *c*, variando em (0,99; 0,95; 0,9; 0,85; 0,75; 0,5). Observou-se que os menores erros de classificação eram obtidos com o conjunto (0,99; 0,95; 0,9; 0,85). A *taxa de mudança da solução atual* para novos vizinhos era obtida do conjunto mutação, descrito anteriormente no método AG. A *temperatura inicial* foi testada para os valores (100; 50; 30; 20; 10; 5; 1; 0,5; 0,1), sendo obtidos menores erros de classificação com o conjunto (1; 0,5; 0,1).

3.5.2 Metodologia de Treinamento

A solução inicial do método de *simulated annealing* é um dos multidiscriminadores previamente armazenados, gerados pelo método tradicional, descrito na Seção 3.3.1. Durante o processo de otimização, a cada iteração, o treinamento de uma nova solução é feito, usando as partições **B0** e **B1** para os multidiscriminadores Buffalo, e **E0** e **E1**, para os multidiscriminadores Essex – utilizando o algoritmo *one-shot-learning* descrito na Seção 2.2.2.

A *quantidade de iterações* testada variou de (6000; 1200; 1000; 700; 600; 500; 450; 400; 200; 150; 100), sendo escolhida 600 iterações. A determinação dos valores para o *esquema de resfriamento*, a *taxa de mudança da solução atual* e a *temperatura inicial* foi feita por uma execução prévia de 50 iterações para cada multidiscriminador a ser otimizado. O melhor trio de valores *esquema de resfriamento* x *taxa de mudança da solução atual* x *temperatura inicial* obtido pela prévia era usado como entrada, para otimizar cada multidiscriminador. Dessa forma, 180 trios foram obtidos automaticamente pela execução prévia, um para cada um dos 30 multidiscriminadores de cada uma das 3 topologias de cada uma das 2 bases.

Levando em consideração todos os procedimentos descritos acima, após 600 iterações, o algoritmo pára e, então, fornece como saída a solução que apresentou o menor valor obtido pela função de custo. Nesse momento, essa solução tem sua memória limpa e, então, é treinada, mais uma vez, pelo algoritmo *one-shot-learning*, descrito na Seção 2.2.2, com o conjunto de treinamento formado pelas partições **B0**, **B1** e **B2**, para os multidiscriminadores Buffalo, e **E0**, **E1** e **E2**, para os Essex.

A mesma observação, feita na Seção 3.4.2 sobre o algoritmo de treinamento ser híbrido e possuir tendência a *overfitting*, também vale para o método SA. Esse algoritmo híbrido é formado pelo algoritmo *one-shot-learning*, descrito na Seção 2.2.2, para treinar os multidiscriminadores e pelo método SA, que solução com uma função de custo menor do que a solução da iteração atual tem maior probabilidade de ser aceita como a próxima solução.

3.5.3 Aspecto Observado

É o mesmo aspecto descrito na Seção 3.3.3.

3.5.4 Resultados Obtidos

Para cada um dos 30 multidiscriminadores de cada topologia gerados na Seção 3.3.1, foram feitas 10 execuções do método SA que tinham um mesmo trio de valores *esquema de resfriamento* x *taxa de mudança da solução atual* x *temperatura inicial*, gerando 10 novos multidiscriminadores para cada um dos 30 multidiscriminadores das três topologias escolhidas e das bases de dados. Então, para cada topologia de cada base de dados, são gerados 300 novos multidiscriminadores (10 execuções x 30 multidiscriminadores). Abaixo, são apresentadas duas tabelas para cada base de dados, contendo a média, o desvio padrão e o melhor resultado do aspecto observado e do *fan-in* médio nesses multidiscriminadores resultantes. Comentários do desempenho são feitos logo em seguida.

BUFFALO

	Topologias		
	T0	**T1**	**T2**
Média	7,97	6,97	21,80
Desvio Padrão	0,77	0,58	1,43
Melhor Resultado	6,82	6,11	18,38

Tabela 3.9 – Erro de classificação das 3 topologias otimizadas pelo método SA, base Buffalo (%).

	Topologias		
	T0	**T1**	**T2**
Média	15,79	11,98	5,97
Desvio Padrão	0,34	0,05	0,06
Melhor Resultado	14,25	11,52	5,71

Tabela 3.10 – *Fan-in* médio das 3 topologias otimizadas pelo método SA, base Buffalo.

Como no caso do método AG, observou-se na Tabela 3.9 que quanto menor a média do EC, menor o desvio padrão. O desvio padrão para cada topologia foi pequeno em relação à média, indicando resultados homogêneos. Todas as três topologias apresentaram diferenças significativas entre as médias dos erros de classificação, para um nível de significância de 5%. Devido à maneira como foi implementada a geração de novos vizinhos, ocorreram poucas desativações das conexões dos neurônios, conforme observado na Tabela 3.10.

ESSEX

	Topologias		
	T0	**T1**	**T2**
Média	11,92	8,73	13,64
Desvio Padrão	1,03	0,75	1,22
Melhor Resultado	9,82	7,26	11,06

Tabela 3.11 – Erro de classificação das 3 topologias otimizadas pelo método SA, base Essex (%).

	Topologias		
	T0	**T1**	**T2**
Média	15,93	11,99	5,92
Desvio Padrão	0,24	0,02	0,13
Melhor Resultado	14,38	11,85	5,36

Tabela 3.12 – *Fan-in* médio das 3 topologias otimizadas pelo método SA, base Essex.

A Tabela 3.11 manteve a tendência de que quanto menor a média, menor o desvio padrão. Essa tabela também apresenta desvios padrões baixos, indicando uma homogeneidade nos erros de classificação dos multidiscriminadores. Todas as três topologias apresentaram diferenças significativas entre as médias dos erros de classificação, para um nível de significância de 5%. Devido à maneira como foi implementada a geração de novos vizinhos, ocorreram poucas desativações das conexões dos neurônios, conforme observado na Tabela 3.12.

3.6 Experimentos com *Tabu Search*

Antes de apresentar os resultados obtidos, usando TS, é importante descrever os procedimentos usados e os elementos principais, conforme descrito na Seção 2.3.3, que são a representação, a função de custo, o método de geração de novos vizinhos e os parâmetros de controle.

3.6.1 Elementos Principais do Algoritmo

Representação, Função de Custo e Geração de Novos Vizinhos

A representação das soluções e a função de custo para o método TS são, respectivamente, a representação e a função de avaliação de aptidão usadas para o método AG, descritas na Seção 3.4.1. A geração de novos vizinhos é a mesma usada pelo método SA, descrita na Seção 3.5.1.

Parâmetros de Controle

Para o método TS, os menores erros de classificação foram obtidos com a *taxa de mudança da solução atual* igual a 0,001. A *quantidade de soluções geradas a cada iteração* foi testada para os valores (30; 20; 15; 10; 5), sendo os melhores resultados obtidos com valor 30. O *tamanho da lista tabu* foi testado para os valores de (30; 10; 5), sendo escolhido o tamanho 5 para a lista.

3.6.2 Metodologia de Treinamento

A solução inicial do método de *Tabu Search* é um dos multidiscriminadores, previamente armazenados, gerados pelo método tradicional descrito na Seção 3.3.1. Durante o processo de otimização, a cada iteração, o treinamento de uma nova solução é feito usando as partições **B0** e **B1** para os multidiscriminadores Buffalo, e **E0** e **E1** para os multidiscriminadores Essex, utilizando o algoritmo *one-shot-learning*, descrito na Seção 2.2.2. A *quantidade de iterações* testada variou de (50; 45; 40; 35; 30; 25; 20; 15; 10; 5), sendo escolhido o valor de 5 iterações.

Levando-se em consideração os procedimentos descritos acima, após 5 iterações, o algoritmo pára, sendo, então, verificada qual foi a solução que apresentou menor valor obtido pela função de custo. Nesse momento, essa solução tem sua memória limpa e, então, é treinada, mais uma vez, pelo algoritmo *one-shot-learning*, descrito na Seção 2.2.2. O conjunto de treinamento é formado pelas partições **B0**, **B1** e **B2**, para os multidiscriminadores Buffalo, e **E0**, **E1** e **E2**, para os Essex.

A mesma observação feita na Seção 3.4.2 sobre o fato de o algoritmo de treinamento ser híbrido e possuir tendência a *overfitting* também vale para o método TS. Esse algoritmo híbrido é formado pelo algoritmo *one-shot-learning*, descrito na Seção 2.2.2, para treinar os multidiscriminadores e pelo método TS, que escolhe, com maior probabilidade para a próxima iteração, soluções que possuam uma função de custo menor que a função de custo da solução da iteração atual.

3.6.3 Aspecto Observado

É o mesmo aspecto descrito na Seção 3.3.3.

3.6.4 Resultados Obtidos

Para cada um dos 30 multidiscriminadores de cada topologia, gerados na Seção 3.3.1, foram feitas 10 execuções do método TS que geraram 10 novos multidiscriminadores para cada um dos 30 multidiscriminadores das três topologias escolhidas e das bases de dados. Então, para cada topologia de cada base de dados, são gerados 300 novos multidiscriminadores (10 execuções x 30 multidiscriminadores). Abaixo, são apresentadas duas tabelas para cada base de dados, contendo a média, o desvio padrão e o melhor resultado do aspecto observado e do *fan-in* médio nesses multidiscriminadores resultantes. Comentários do desempenho são feitos logo em seguida.

BUFFALO

	Topologias		
	T0	**T1**	**T2**
Média	7,53	6,87	20,64
Desvio Padrão	0,46	0,42	1,10
Melhor Resultado	6,21	5,96	17,68

Tabela 3.13 – Erro de classificação das 3 topologias otimizadas pelo método TS, base Buffalo (%).

	Topologias		
	T0	**T1**	**T2**
Média	15,95	11,98	5,99
Desvio Padrão	0,06	0,03	0,02
Melhor Resultado	15,61	11,80	5,93

Tabela 3.14 – *Fan-in* médio das 3 topologias otimizadas pelo método TS, base Buffalo.

A Tabela 3.13 manteve a tendência de que quanto menor a média, menor o desvio padrão. Todas as três topologias apresentaram diferenças significativas entre as médias dos erros de

classificação, para um nível de significância de 5%. Devido à maneira como foi implementada a geração de novos vizinhos, ocorreram poucas desativações das conexões dos neurônios, conforme observado na Tabela 3.14.

ESSEX

	Topologias		
	T0	**T1**	**T2**
Média	12,15	9,20	13,08
Desvio Padrão	0,89	0,65	0,91
Melhor Resultado	9,65	7,35	10,80

Tabela 3.15 – Erro de classificação das 3 topologias otimizadas pelo método TS, base Essex (%).

	Topologias		
	T0	**T1**	**T2**
Média	15,96	11,98	5,98
Desvio Padrão	0,05	0,02	0,02
Melhor Resultado	15,70	11,86	5,83

Tabela 3.16 – *Fan-in* médio das 3 topologias otimizadas pelo método TS, base Essex.

O desvio padrão das topologias otimizadas também manteve a tendência em que quanto menor a média, menor o desvio padrão. Todas as três topologias apresentaram diferenças significativas entre as médias dos erros de classificação, para um nível de significância de 5%. Devido à maneira como foi implementada a geração de novos vizinhos, ocorreram poucas desativações das conexões dos neurônios, conforme observado na Tabela 3.16.

3.7 Considerações Finais

As topologias, avaliadas no método tradicional, permitiram tirar duas conclusões importantes:

- Quanto maior o número de neurônios, mantendo o *fan-in* constante, menor é o EC. Todavia, o tempo de treinamento e o tempo de teste são diretamente proporcionais à quantidade de neurônios. Esse fato é relevante, ao utilizar métodos de otimização global, em que o treinamento e o teste são feitos milhares de vezes, até se obter uma saída.
- Quanto maior o *fan-in*, mantendo o número de neurônios constante, menor é o EC, se houver número de exemplos significativos, o suficiente para evitar superespecialização. Caso contrário, o EC tende a aumentar após um determinado tamanho de *fan-in*.

Em relação aos erros médios de classificação e ao *fan-in* médio, obtidos para cada topologia de cada método, eles foram bastante homogêneos. Apenas, no método tradicional, ocorreram seis topologias que não apresentaram diferenças significativas nos erros médios de classificação, ao nível de significância de 5%.

Também foi observada uma tendência a *overfitting* da metodologia de treinamento híbrida para a partição **B2** e **E2**, respectivamente, para os multidiscriminadores Buffalo e Essex. Isso aconteceu porque multidiscriminadores mais adaptados a reconhecer padrões dessas partições são obtidos pela forma como a metodologia foi aplicada.

Por fim, em relação à base Buffalo, o tempo médio gasto, em um Pentium IV de 1.7GHz com 512 MB de RAM, para otimizar a topologia T0 era de, aproximadamente, um dia e meio, para otimizar a topologia T1 era de, aproximadamente, três dias e para otimizar a topologia T2 era de, aproximadamente, cinco dias. Em relação à base Essex, devido ao menor número de padrões, o tempo médio reduziu-se em, aproximadamente, 40%. O tempo médio não apresentou variações significativas entre os três métodos de otimização global.

Capítulo 4

4 Análises Comparativas

4.1 Introdução

Este capítulo apresenta um estudo comparativo dos resultados obtidos pelos diversos métodos usados nesta dissertação – o tradicional, o algoritmo genético, o *simulated annealing* e o *tabu search* – com as bases de dados Buffalo e Essex. Essa análise é baseada nos resultados do erro de classificação, obtidos pelas três topologias iniciais escolhidas na seção 3.3.5.

4.2 Análise Comparativa com a Base Buffalo

Para uma melhor análise comparativa, três tabelas resumidas são apresentadas. A primeira tabela resume os resultados obtidos no Capítulo 3; a segunda resume os resultados obtidos pelos testes de hipótese, levando em consideração um nível de significância de 5% e a terceira tabela mostra a média e o desvio padrão de conexões dos neurônios dos multidiscriminadores.

Métodos	Topologias								
	T0			**T1**			**T2**		
	Média	Desvio	Melhor	Média	Desvio	Melhor	Média	Desvio	Melhor
Tradicional	7,52	0,40	6,82	6,85	0,38	6,11	24,95	1,21	22,78
Algoritmos Genéticos	7,44	0,42	6,41	7,08	0,39	6,31	23,99	0,79	22,68
Simulated Annealing	7,97	0,77	6,82	6,97	0,58	6,11	21,80	1,43	18,38
Tabu Search	7,53	0,46	6,21	6,87	0,42	5,96	20,64	1,10	17,68

Tabela 4.1 – Taxas de erro de classificação, base Buffalo (%).

Comparativos	Topologias		
	T0	**T1**	**T2**
Tradicional x AG	Sem diferença significativa	Tradicional	AG
Tradicional x SA	Tradicional	Sem diferença significativa	SA
Tradicional x TS	Sem diferença significativa	Sem diferença significativa	TS
AG x SA	AG	Sem diferença significativa	SA
AG x TS	Sem diferença significativa	TS	TS
SA x TS	TS	Sem diferença significativa	TS

Tabela 4.2 – Resumo dos comparativos obtidos pelo teste de hipótese (α = 5%), base Buffalo.

Comparativos	Topologias					
	T0		**T1**		**T2**	
	Média	Desvio	Média	Desvio	Média	Desvio
Tradicional	16,00	–	12,00	–	6,00	–
AG	14,66	0,58	11,33	0,29	5,31	0,13
SA	15,79	0,34	11,98	0,05	5,97	0,06
TS	15,95	0,06	11,98	0,03	5,99	0,02

Tabela 4.3 – Resumo das média das conexões por neurônio, base Buffalo.

Cada coluna da Tabela 4.2 apresenta os resultados obtidos pelo teste estatístico de hipótese comparando cada par de método, indicando se não há diferença significativa entre eles, obtida pelo teste bicaudal ou qual dos métodos foi superior, obtido pelos testes unicaudais, considerando um nível de significância de 5%.

4.2.1 Comparativos da Topologia T0

A Tabela 4.2 mostra que existem evidências estatísticas de que os métodos tradicional, AG e TS apresentaram soluções com EC sem diferenças significativas. Por outro lado, o método AG conseguiu uma diminuição do *fan-in* médio de 16 para 14,66, reduzindo a memória ocupada em 61% ($2^{16} \times 64 \times 10 = 41943040 = 40MB$, enquanto que $2^{14,66} \times 64 \times 10 \cong 16568367 \cong 15,8MB$). Isso torna o uso desse método o mais apropriado para otimizar essa topologia. Essa diminuição na quantidade de conexões com melhora na performance indica que a topologia de 64 neurônios com 16 conexões está superdimensionada para esse problema.

4.2.2 Comparativos da Topologia T1

A Tabela 4.2 mostra que os métodos tradicional, SA e TS apresentam multidiscriminadores com EC sem diferenças significativas. Já os EC obtidos pelos multidiscriminadores otimizados por AG são significantemente inferiores que os obtidos pelos métodos tradicional e TS e sem diferenças significativas aos obtidos por SA.

Esse aumento no EC médio do método AG pode ter sido pequeno (0,23%), embora tenha sido significante. Uma possível causa para o comportamento do método AG pode ser atribuída a sua maior tendência a desativar conexões, devido à maneira como foi implementado. Pois, enquanto que o método AG chegou a desativar quase 8,5% do *fan-in* médio, o método SA desativou menos de 1,5%, e o método TS, menos de 0,5%. Com o método AG, resultando em multidiscriminadores com padrões de conectividade menores, aumenta-se a probabilidade de esse método gerar multidiscriminadores com padrões de conectividades mais adaptados às

características específicas do particionamento **B2**. Isso se torna mais evidente quando o *fan-in* da topologia inicial for próximo do ótimo. Dessa forma, a capacidade de generalização dos multidiscriminadores diminui, levando o método AG a gerar multidiscriminadores com EC significantemente maiores do que os obtidos pelo método tradicional.

4.2.3 Comparativos da Topologia T2

A Tabela 4.2 mostra que os EC dos multidiscriminadores, obtidos pelos métodos de otimização global, são significantemente menores que os obtidos pelo método tradicional. Vale ressaltar que a diferença entre as médias do método tradicional e TS foi de 4,31%, indicando uma melhora de 17,27% no valor de erro de classificação médio ($EC_{tradicional} = 24{,}95\%, EC_{TS} = 20{,}64\% \Rightarrow EC_{TS} = 82{,}72\% EC_{tradicional}$).

Uma das razões para a superioridade dos métodos de otimização global em relação ao método tradicional se deve à forma como o método tradicional gera o padrão de conectividade. O padrão de conectividade, no método tradicional, é criado de forma uniformemente aleatória. Sendo assim, todos os bits do vetor de entrada terão a mesma probabilidade de serem conectados aos neurônios dos discriminadores, independentemente da relevância desses bits na tarefa de determinar a classe do padrão de entrada. Além disso, o conjunto de bits do vetor de entrada relevante para discriminar uma classe tende a variar pouco nos padrões da classe e, em contrapartida, os bits do vetor de entrada pouco relevantes não possuem tendência definida. Sendo assim, se o neurônio estiver conectado a um conjunto de bits pouco relevante para a classe de seu discriminador, o comportamento desses bits provocará ruído na memória do neurônio. Esse ruído provocará o acesso de uma posição de memória diferente a cada novo padrão aprendido. Além disso, quanto maior for o vetor de entrada, maior é a quantidade de bits pouco relevantes a determinadas classes nesse vetor. Assim, se o tamanho dessa memória for pequeno, o neurônio ficará saturado rapidamente, e esses neurônios saturados acabam provocando ruído nos seus discriminadores.

Dessa forma, na topologia **T2** inicial, para o caso do método tradicional, um neurônio possui um *fan-in* de 6, indicando que ele pode produzir 64 respostas diferentes para um mesmo

conjunto de conexões. Como o conjunto de treinamento, formado pelas partições **B0**, **B1** e **B2**, possui 594 padrões por classe, se o neurônio estiver ligado a um conjunto de bits que tenha uma alta variabilidade, a maior parte das suas 64 posições será preenchida com 1, saturando os neurônios e degradando a performance da rede.

Nos métodos de otimização global, o padrão de conectividade dos multidiscriminadores é flexível no sentido que ele é redistribuído de modo a minimizar o erro de classificação da partição **B2**. Isso tende a evitar, por exemplo, que os neurônios sejam conectados a partes do padrão que levem a uma saturação destes, melhorando a performance da rede. Também vale observar que o método AG possui uma maior probabilidade de desativar conexões do que os métodos SA e TS, gerando neurônios com um *fan-in* menor que 6. Ao fazer isso, AG obtém neurônios com muito poucas posições de memórias (32 ou menos), tendo uma chance maior do que os métodos SA e TS de saturá-las.

Ao levar em consideração os argumentos expostos acima, percebe-se por que os métodos de otimização global obtiveram EC médios significantemente menores em um nível de significância de 5% que o método tradicional e, ainda, por que o método AG obteve EC médios significantemente maiores que os métodos SA e TS.

4.3 Análise Comparativa com a Base Essex

Da mesma forma realizada para a base Buffalo, para uma melhor análise comparativa do comportamento dos métodos na base Essex, três tabelas resumidas são apresentadas. A primeira tabela resume os resultados obtidos no Capítulo 3; a segunda resume os resultados obtidos pelos testes de hipótese, levando em consideração um nível de significância de 5% e a terceira tabela mostra a média e o desvio padrão de conexões dos neurônios dos multidiscriminadores.

Métodos	Topologias								
	T0			**T1**			**T2**		
	Média	Desvio	Melhor	Média	Desvio	Melhor	Média	Desvio	Melhor
Tradicional	11,58	0,54	10,18	8,45	0,45	7,26	15,25	1,00	13,63
Algoritmos Genéticos	11,37	1,09	9,73	8,71	0,57	7,26	13,83	0,92	12,48
Simulated Annealing	11,92	1,03	9,82	8,73	0,75	7,26	13,64	1,22	11,06
Tabu Search	12,15	0,89	9,65	9,20	0,65	7,35	13,08	0,91	10,80

Tabela 4.4 – Taxas de erro de classificação, base Essex (%).

Comparativos	Topologias		
	T0	**T1**	**T2**
Tradicional x AG	Sem diferença significativa	Tradicional	AG
Tradicional x AS	Tradicional	Tradicional	SA
Tradicional x TS	Tradicional	Tradicional	TS
AG x SA	AG	Sem diferença significativa	Sem diferença significativa
AG x TS	AG	AG	TS
SA x TS	Sem diferença significativa	SA	TS

Tabela 4.5 – Resumo dos comparativos obtidos pelo teste de hipótese (α = 5%), base Essex.

Comparativos	Topologias					
	T0		**T1**		**T2**	
	Média	Desvio	Média	Desvio	Média	Desvio
Tradicional	16,00	–	12,00	–	6,00	–
AG	12,89	0,74	11,35	0,39	5,27	0,16
SA	15,93	0,24	11,99	0,02	5,92	0,13
TS	15,96	0,05	11,98	0,02	5,98	0,02

Tabela 4.6 – Resumo das média das conexões por neurônio, base Essex.

Analogamente à Tabela 4.2, cada coluna da Tabela 4.5 apresenta os resultados obtidos pelo teste estatístico de hipótese, comparando cada par de método, indicando se não há diferença significativa entre eles obtida pelo teste bicaudal ou qual dos métodos foi superior, obtido pelos testes unicaudais, considerando um nível de significância de 5%.

4.3.1 Comparativos da Topologia T0

Como aconteceu na Seção 4.2.1, os EC dos multidiscriminadores, obtidos pelo método AG, não apresentaram diferença significativa aos identificados pelo método tradicional, mas a memória ocupada pela arquitetura foi reduzida, nesse caso, em 89%, ao reduzir o *fan-in* médio de 16 para 12,89, tornando, novamente, o método AG o mais apropriado para a topologia **T0** ($2^{16} \times 64 \times 10 = 41943040 = 40MB$, enquanto que $2^{12,89} \times 64 \times 10 \cong 4857990 \cong 4{,}6MB$). Essa diminuição na quantidade de conexões, aliada a uma diminuição do erro médio de classificação, indica que a arquitetura estava superdimensionada. Outro indício desse superdimensionamento é verificado ao observar que os multidiscriminadores da topologia de 64 neurônios com 12 conexões, obtidos pelo método tradicional, possuem erros de classificação médio menor do que o da topologia de 64 neurônios com 16 conexões (Ver resultados na Tabela 3.3).

Outro fato relevante é que houve tendência de *overfitting* dos métodos de otimização global aos padrões da partição **E2**, pois os métodos SA e TS apresentaram multidiscriminadores com EC maiores do que o método tradicional. Para o método AG, essa tendência também ocorre, se compararmos os EC médios com aqueles obtidos pelo método tradicional para a topologia 64 neurônios com 12 conexões, que é a topologia mais próxima da topologia média, obtida por AG (64 neurônios com 12,89 conexões).

4.3.2 Comparativos da Topologia T1

Essa foi a única topologia, em que os métodos de otimização não acharam multidiscriminadores com EC menores que aqueles obtidos pelos multidiscriminadores, gerados com o método tradicional.

A razão desse comportamento é a tendência de *overfitting* dos métodos de otimização global à partição **E2** (113 padrões por classe). A tendência de *overfitting* é maior do que a apresentada na Seção 4.2.2, devido a um menor número de padrões da partição **E2**, já que a partição **B2** possui 75% mais padrões. Com isso, os métodos de otimização global tenderam a superdiscriminar.

4.3.3 Comparativos da Topologia T2

A Tabela 4.5 mostra que todos os métodos de otimização global produziram multidiscriminadores com EC significantemente menores que os obtidos pelos multidiscriminadores, gerados com o método tradicional. Nessa tabela, mostra-se que TS obteve multidiscriminadores com EC significantemente menor do que AG e SA. Também mostra que os EC dos multidiscriminadores, gerados por SA, não apresentam diferenças significativas aos EC dos multidiscriminadores, gerados pelo método AG. A diferença entre as médias dos EC para o método tradicional e TS na Tabela 4.5 foi de 2,17%, indicando uma melhora de 14,23% no valor da discriminação ($EC_{tradicional} = 15{,}25\%, EC_{TS} = 13{,}08\% \Rightarrow EC_{TS} = 85{,}77\% EC_{tradicional}$).

A razão para os métodos de otimização global terem gerado multidiscriminadores apresentando menores EC do que os EC obtidos pelos multidiscriminadores do método tradicional é análoga à Seção 4.2.3. A principal diferença é que, como o conjunto de treinamento formado pelas partições **E0**, **E1** e **E2** é 43% menor do que o conjunto formado por **B0**, **B1** e **B2**, o que ocasionou uma menor saturação nas memórias dos multidiscriminadores obtidos pelo método AG. Como conseqüência, os EC dos multidiscriminadores, obtidos por AG, não apresentaram diferenças significativas aos obtidos por SA. Essa menor saturação também acarretou em uma menor diferença, em comparação às diferenças obtidas na Seção 4.2.3, entre os EC médios obtidos pelos métodos de otimização global e o EC médio obtido pelo método tradicional.

4.4 Considerações Finais

Ao se compararem os resultados da topologia **T1**, independentemente das bases de dados, os métodos de otimização global não geraram soluções com desempenhos significantemente superiores ao método tradicional em nenhum dos seis testes (Tradicional x AG, Tradicional x SA e Tradicional x TS para cada base). Para essa topologia, apenas em dois casos – SA e TS para a base Buffalo –, a média dos erros de classificação apresentou diferenças significativas às obtidas pelo método tradicional. Mas, em todos os outros quatro casos, os métodos de otimização global geraram soluções com performance significantemente inferiores ao método tradicional, considerando um nível de significância de 5%. Por isso, de um modo geral, para essa topologia, os erros de classificação obtidos indicam que os métodos de otimização global não são apropriados.

Isso ocorreu devido à metodologia de treinamento híbrida, adotada pelos métodos de otimização global, que possui uma tendência de *overfitting*, obtendo soluções mais adaptadas às partições **B2** e **E2**, respectivamente, para o contexto da base Buffalo e Essex. No contexto da base Essex, o menor número de padrões na partição **E2** aumentou a tendência de *overfitting*, fazendo os métodos de otimização global gerar multidiscriminadores com tendência a superdiscriminar. Já, no contexto da base Buffalo, o maior número de padrão na partição **B2** diminuiu essa tendência, exceto para o método AG, por gerar multidiscriminadores com um *fan-in* menor do que SA e TS. Sendo assim, para essa topologia, da forma como foram desenvolvidos os experimentos, o método tradicional se mostra mais apropriado que os métodos de otimização global.

Em relação à topologia **T0**, AG obteve multidiscriminadores com EC sem diferença significativa aos do método tradicional, embora, com memória alocada, reduzida em 61%, no contexto da base Buffalo e, em 89%, no contexto da base Essex. Portanto, o método AG, além de otimizar o padrão de conectividade dos neurônios, também consegue otimizar o número de conexões. Essa redução ocorreu com os multidiscriminadores otimizados pelo método AG devido à forma como o método foi implementado. Ao reduzir as conexões e gerar

multidiscriminadores com EC sem diferença significativa aos obtidos pelo método tradicional, conclui-se que esta arquitetura de 64 neurônios com 16 conexões é superdimensionada para as duas bases de dados. Sendo assim, nessa topologia, o método AG se mostrou mais apropriado que o método tradicional, ao encontrar multidiscriminadores com EC médio sem diferenças significativas, embora, com uma redução significativa na memória alocada.

Em relação à topologia **T2**, todos os métodos de otimização global apresentaram multidiscriminadores com EC significantemente menores que o método tradicional, em um nível de significância de 5%. Com esta topologia, percebeu-se que os multidiscriminadores obtidos pelo método tradicional não eram boas soluções, pois os padrões de conectividade dos neurônios com *fan-in* pequeno eram dispersos uniformemente, aumentando as chances de gerar vários neurônios com poucas conexões relevantes. Por exemplo, neurônios com uma memória de 64 posições e conectados a bits com alta variabilidade para a classe, acrescendo-se, ainda, a um número grande de padrões no conjunto de treinamento em relação à quantidade de memória (339 no contexto da base Essex e 594 no contexto da base Buffalo), acarreta uma rápida saturação dos neurônios, degradando a performance do multidiscriminador.

Por outro lado, ao utilizar os métodos de otimização global, o padrão de conectividade dos neurônios era rearranjado de tal maneira que os bits mais relevantes dos padrões de entrada do conjunto de treinamento de cada classe fossem mais conectados aos neurônios do discriminador da classe, melhorando, sensivelmente, a performance dos multidiscriminadores. O método TS obteve os menores erros de classificação, reduzindo o erro de classificação do método tradicional em 17,27% no contexto da base de dados Buffalo e em 14,23%, no contexto da base de dados Essex. Então, para essa topologia, o método TS é o mais apropriado, pois obteve os multidiscriminadores com os menores EC, considerando um nível de significância de 5%.

É importante observar que o uso ou não de métodos de otimização global irá depender dos objetivos pretendidos. Se houver uma metodologia, para selecionar o melhor modelo, os métodos de otimização global podem ser um complemento ao método tradicional. Em especial, o método TS, pois ele obteve o menor erro de classificação em quase todas as três topologias nas duas bases de dados. No único caso em que TS obteve erro de classificação maior do que o método tradicional, tampouco AG e SA conseguiram encontrar menor erro de classificação que o método

tradicional. Se o objetivo for gerar um multidiscriminador que aloque menor espaço de memória, o método AG é o mais apropriado, pois, da forma como foi implementado, gera soluções com menos conexões do que SA e TS.

Capítulo 5

5 Objetivos Alcançados e Trabalhos Futuros

5.1 Introdução

O uso integrado de técnicas distintas, para formar sistemas mais robustos, é um dos principais objetivos atuais da área de Inteligência Artificial. Sua motivação reside no fato de que técnicas diferentes podem ser adequadas para alguns casos, embora, deficientes para outros. Com o intuito de superar as desvantagens individuais, combinam-se as técnicas, com o objetivo de aproveitar as vantagens de cada uma. [LUDERMIR et al., 2003]

Por exemplo, neste trabalho, três técnicas de otimização global foram utilizadas para otimizar o padrão de conectividades de redes neurais, com o objetivo de investigar o desempenho das redes resultantes e de diminuir a aleatoriedade do processo de geração dos multidiscriminadores com o método tradicional. Também foi de interesse utilizar um problema do mundo real, e não, apenas, problemas de fácil resolução, como é de praxe na literatura.

5.2 Objetivos Alcançados

No Capítulo 3, foi descrito o problema do mundo real usado por este trabalho, o reconhecimento de caracteres numéricos manuscritos. Também foi descrito como os dados foram particionados, de modo a evitar qualquer tipo de viés (qualquer tendência amostral sem correspondência populacional), pressuposto necessário para garantir que o teste estatístico de hipótese, explicado na Seção 3.2, fosse o mais robusto possível. Para isso, foi implementada uma forma de carregar todos os padrões em um banco de dados e de particioná-los.

Também no Capítulo 3, foi criada uma representação de multidiscriminadores, necessária para a utilização dos métodos de otimização global. Esta representação permitia uma codificação e decodificação rápida e precisa dos multidiscriminadores. Nesse mesmo capítulo, também foram desenvolvidas maneiras de manipular soluções codificadas, de modo a gerar novas soluções possíveis. Além disso, procedimentos, para otimizar multidiscriminadores, foram descritos para cada metodologia. Os parâmetros desses procedimentos eram escolhidos para cada topologia ou multidiscriminador, dependendo do método de otimização global, automaticamente por meio de execuções prévias em que eram testadas várias variações. Tudo isso foi implementado, utilizando C#.Net© com todos os multidiscriminadores, sendo armazenados em um servidor de banco de dados MS SQL Server 2000©.

No Capítulo 4, foi feita uma análise comparativa entre os métodos tradicional, Algoritmos Genéticos, *Simulated Annealing* e *Tabu Search*, indicando se houve alguma diferença significativa de performance entre eles, como também uma discussão sobre os resultados obtidos. Ao final, alguns comentários foram feitos, indicando em que casos os métodos seriam mais apropriados. Todos os resultados obtidos foram compilados e resumidos em planilhas MS Excel© bem como os comparativos feitos pelos testes de hipótese.

5.3 Trabalhos Futuros

Uma das possíveis pesquisas futuras que este trabalho pode motivar é a de investigar se o comportamento dos métodos de otimização global com outras topologias reagem, conforme as conclusões obtidas no Capítulo 4.

Outra sugestão que deve melhorar a performance dos métodos de otimização global é o uso de um conjunto de validação durante o processo de otimização, para diminuir a tendência de *overfitting* dos multidiscriminadores, ao utilizar a metodologia de treinamento híbrida.

Já, em relação ao método TS, uma sugestão seria utilizar um procedimento de geração de novos vizinhos, análogo ao operador de mutação do método AG, para ter maior probabilidade de desativar conexões dos neurônios. Nesse mesmo contexto, também poderia ser investigado o desempenho do método AG, ao usar o operador de mudança dos métodos SA e TS como operador de mutação, o que acarretaria para o método AG uma menor tendência para desativar conexões.

Por fim, como o foco deste trabalho era verificar a performance dos métodos de otimização global (Algoritmos Genéticos, *Simulated Annealing* e *Tabu Search*), poucas modificações foram feitas nesses métodos de otimização. Diversas sugestões de modificações nos métodos são feitas na literatura [BRANKE, 1995] [PHAM e KARABOGA, 2000] [SEXTON et al., 1998] [YAO, 1999], de forma a melhorar seus desempenhos. Como mais uma sugestão para um trabalho futuro, acredito que a prática de alguma dessas modificações venha a produzir resultados melhores aos obtidos por este trabalho.

Bibliografia

[ALEKSANDER, 1966] I. Aleksander. *Self-adaptive universal logic circuits*. Electronic Letters 2, p. 231, 1966.

[ALEKSANDER, 1983] I. Aleksander. *Emergent intelligent properties of progressively structured pattern recognition nets*. Pattern Recognition Letters, 1: pp. 375-384, 1983.

[ALEKSANDER, 1989] I. Aleksander. *Canonical neural nets based on logic nodes*. In Proceedings of IEE International Conference on Artificial Neural Network, London, pp. 110-114, 1989.

[ALEKSANDER e MORTON, 1991] I. Alexander e H. Morton. *General Neural Unit: Retrieval Performance*. IEE Eletronics Letters, 27: 1776-1778, 1991.

[ALEKSANDER e STONHAM, 1979] I. Aleksander e T. J. Stonham. *A guide to pattern recognition using random access memories*. IEE Journal Computers and Digital Techniques 2(1), pp. 29-40, 1979.

[ALEKSANDER et al., 1984] I. Aleksander, W. V. Thomas e P. A. Bowden. *WISARD: a radical step forward in image recognition*. Sensor Review 4(3), pp. 120-124, 1984.

[BAXTER, 1992] J. Baxter. *The Evolution of Learning Algorithms for Artificial Neural Networks*. Em Complex Systems, D. Green and T. Bossomaier, eds., pp. 313-326, IOS Press, Amsterdam, 1992.

[BISHOP et al., 1990] J. M. Bishop, A. A. Crowe, P. R. Minchinton e R. J. Mitchell. *Evolutionary Learning to Optimize Mapping in n-Tuple Networks*. IEE Colloquium on "Machine Learning", Digest 1990, p. 117, 1990.

[BLEDSOE e BROWNING, 1959] W. W. Bledsoe e I. Browning. *Pattern recognition and reading by machine*. Proceedings of the Eastern Joint Computer Conference, pp. 225-232, 1959.

[BRAGA et al., 2000] A. P. Braga, T. B. Ludermir, A. C. P. L. F. Carvalho. *Redes Neurais Artificiais – Teoria e aplicações*. Pp. 131-146; 161-170. LTC – Livros Técnicos e Científicos Editora S.A., 2000.

[BRANKE, 1995] J. Branke. *Evolutionary Algorithms for Neural Network Design and Training*. Technical Report n. 322, Institute AIFB, University of Karlsruhe, janeiro de 1995.

[CANUTO, 2001] A. M. P. Canuto. *Combining Neural Networks and Fuzzy Logic for Applications in Character Recognition*. Doctor's thesis, The University of Kent at Canterbury, England. Pp. 37-49 e 91-93, maio de 2001.

[CARVALHO et al., 2003] A. C. P. L. F. Carvalho, A. P. Braga e T. B. Ludermir. *Computação Evolutiva*. Em Sistemas Inteligentes – Fundamentos e Aplicações, pp. 227-239. Editora Manole Ltda., 1ª edição, 2003.

[CHALUP e MAIRE, 1999] S. Chalup e F. Maire, *A Study on Hill Climbing Algorithms for Neural Network Training*. In Proceedings of the 1999 Congress on Evolutionary Computation (CEC'99), July 6-9, 1999, Washington, D.C., USA, vol. 3, pp. 2014-2021, 1999.

[DAVIS, 1991] L. Davis. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, NY, 1991.

[DUDA et al., 2000] R. O. Duda, P. E. Hart e D. G. Stork. *Hypothesis Testing*. In Pattern Classification, 2nd ed., pp. 628-630, John Wiley & Sons Inc., 2000.

[GLOVER, 1986] F. Glover. *Future paths for integer programming and links to artificial intelligence*. In Computers and Operation Research, vol. 13, pp. 533-549, 1986.

[GORSE e TAYLOR, 1990] D. Gorse e J. G. Taylor. *Reinforcement training strategies for probabilistic RAM*. In M. Novak and E. Pelikan (eds.), Proceedings of International Symposium on Neural Networks and Neurocomputing (NEURONET90), pp. 180-184, 1990.

[GORSE e TAYLOR, 1991] D. Gorse e J. G. Taylor. *A continuous input RAM-based stochastic neural model*. Neural Networks 4, pp. 657-665. 1991.

[HANSEN, 1986] P. Hansen. *The steepest ascent mildest descent heuristic for combinatorial programming*. Conference on Numerical Methods in Combinatorial Optimization, Capri, Itália, 1986.

[HOLLAND, 1992] J. H. Holland. *Adaptation in Natural and Artificial Systems (2nd ed.)*. University of Michigan Press, Ann Arbor, MI, 1992.

[HULL, 1994] J. J. Hull. *A database for handwritten text recognition*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 5(16):pp. 550-554. 1994.

[IRIDIANTECH] http://www.iridiantech.com/

[JONG, 1980] K. Jong. *Adaptative System Design: a Genetic Approach*. IEEE Transactions on Systems, Man and Cybernetics SMC-10(9). 1980.

[KAN e ALEKSANDER, 1987] W. K. Kan e I. Aleksander. *A probabilistic logic neuron network for associative learning*. In Proceedings of the IEEE International Conference on Neural Networks, Volume II, San Diego, CA, pp. 541-548, junho de 1987.

[KIRKPATRICK et al., 1983] S. Kirkpatrick, C. D. Gellat Jr. e M. P. Vecchi. *Optimization by simulated annealing*. Science, 220: pp. 671-680, 1983.

[LEE et al., 1997] L. L. Lee, M. G. Lizarraga, N. R. Gomes e A. L. Koerich. *A Prototype For Brazilian Bank check Recognition*. International Journal of Pattern Recognition and Artificial Intelligence, Volume 11, Número 4, pp. 549-569, junho de 1997.

[LEVINE e BERENSON, 2000] D. M. Levine e M. L. Berenson. *Estatística: Teoria e Aplicações – Usando Microsoft Excel Português*. Pp. 283-383. Editora LTC, 1ª edição, 2000.

[LUDERMIR et al., 1999] T. B. Ludermir, A. C. P. L. F. Carvalho, A. P. Braga e M. C. P. Souto. *Weightless Neural Models: A Review of Current and Past Works*. Em Neural Computing Surveys 2, pp. 41-61, 1999.

[LUDERMIR et al., 2003] T. B. Ludermir, A. C. P. L. F. Carvalho, A. P. Braga e M. C. P. Souto. *Sistemas Inteligentes Híbridos*. Em Sistemas Inteligentes – Fundamentos e Aplicações, pp. 254-257. Editora Manole Ltda., 2003.

[MCCULLOCH e PITTS, 1943] W. S. McCulloch e W. Pitts. *A logical calculus of the ideas immanent in nervous activity*. Bulletin of Mathematical Biophysics 5, pp. 115-137, 1943.

[METROPOLIS et al., 1953] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller e E. Teller. *Equation of state calculations by fast computing machines*. Journal of Chem. Phys., volume 21, nº 6, pp. 1087-1092, 1953.

[MICHALEWICZ, 1992] Z. Michalewicz. *Genetic Algorithms + Data Strutures = Evolution Programs*. Springer-Verlag, New York, NY, 1992.

[MYERS e ALEKSANDER, 1988] C. Myers e I. Aleksander. *Learning algorithms for probabilistic logic nodes*. In Abstracts of I Annual INNS Meeting, Boston, p. 205, 1988.

[OSMAN, 1991] I. H. Osman. *Metastrategy Simulated Annealing and Tabu Search Algorithms for Combinatorial Optimisation Problems*, PhD Thesis, University of London, Imperial College, UK, 1991.

[PHAM e KARABOGA, 2000] D. T. Pham e D. Karaboga. *Introduction*. Em D. T. Pham and D. Karaboga (eds.), Intelligent Optimisation Techniques, pp. 1-50, Springer-Verlag, 2000.

[PRADOS, 1992] D. L. Prados. *Training Multilayered Neural Networks by Replacing the Least Fit Hidden Neurons*. Em Proc. of IEEE Southeastcon '92, vol. 2, pp. 634-637, IEEE Press, New York, NY, 1992.

[PRECHELT, 1994] L. Prechelt. *PROBEN 1 – a set of benchmarks and benchmarking rules for neural network training algorithms*. Technical report, Fakultät für Informatik, Universität Karlsruhe. 1994.

[PRECISEBIOMETRICS] http://www.precisebiometrics.com/

[RUMELHART et al., 1986] D. E. Rumelhart, G. E. Hinton e R. J. Williams. *Learning Internal Representations by Error Propagation*. Em D. E. Rumelhart, J. L. McClelland e o PDP Research Group (Eds.), Parallel Distributed Processing, Volume 1, pp. 318-362. Cambridge, MA: MIT Press. 1986.

[SEXTON et al., 1998] R. S. Sexton, B. Alidaee, R. E. Dorsey, J. D. Johnson. *Global Optimization for Artificial Neural Networks: A Tabu Search Application*. No European Journal of Operational Research 106, pp. 570-584. 1998

[SOUTO, 1999] M. C. P. de Souto. *Computability and Learnability in Sequential Weightless Neural Networks*, pp. 16-22 e 72-76. Tese de doutorado, Department of Electrical Engineering, Imperial College of Science, Technology and Medicine, the University of London, julho de 1999.

[STONHAM, 1984] T. J. Stonham. *Pratical Pattern Recognition*. In Advanced Digital Information Systems, pp. 259-271, Prentice-Hall International, 1984.

[STORK e ALLEN, 1992] D. G. Stork e J. D. Allen. *How to solve the n-bit parity problem with two hidden units*. In Neural Networks, 5: pp. 923-926, 1992.

[WATTA et al., 1996] P. Watta, B. Desaie, N. Dannug e M. Hassoun. *Image Compression Using Backprop*. In

http://neuron.eng.wayne.edu/bpImageCompression9PLUS/bp9PLUS.html, 1996.

[WINSTON, 1992] P. Winston. *Artificial Inteligence*. Addison-Wesley. 1992.

[YAMAZAKI, 2001] A. Yamazaki. *Reconhecimento de Padrões em um Nariz Artificial por Redes Neurais*. Dissertação de Mestrado, Centro de Informática, Universidade Federal de Pernambuco, Brasil. Pp. 61-79, junho de 2001.

[YAMAZAKI et al., 2002] A. Yamazaki, T. B. Ludermir e M. C. P. de Souto. *Global Optimization Methods for Designing and Training Neural Networks*. VII Brazilian Symposium on Neural Networks, pp. 130-135, Recife, 11-14 de novembro de 2002.

[YAO, 1999] X. Yao. *Evolving Artificial Neural Networks*. Proceedings of the IEEE, 87(9): pp. 1423-1447, September, 1999.