COPPE/UFRJ

RECONHECIMENTO DE PADRÕES E REDES NEURAIS ARTIFICIAIS EM PREDIÇÃO DE ESTRUTURAS SECUNDÁRIAS DE PROTEÍNAS

Emerson Cordeiro Morais

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Sistemas e Computação.

Orientador: Rubem Pinto Mondaini

Rio de Janeiro
Março de 2010

RECONHECIMENTO DE PADRÕES E REDES NEURAIS ARTIFICIAIS EM PREDIÇÃO DE ESTRUTURAS SECUNDÁRIAS DE PROTEÍNAS

Emerson Cordeiro Morais

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

_____

Prof. Rubem Pinto Mondaini, D.Sc.

_____

Prof. Diego Gervasio Frías Suárez, D.Sc.

_____

Prof. Eduardo Massad, D.Sc.

_____

Prof. Ricardo Cordeiro de Farias, Ph.D.

_____

Prof. Ricardo de Lima Zollner, D.Sc.

RIO DE JANEIRO, RJ - BRASIL
MARÇO DE 2010

Morais, Emerson Cordeiro

Reconhecimento de Padrões e Redes Neurais Artificiais em Predição de Estruturas Secundárias de Proteínas / Emerson Cordeiro Morais. – Rio de Janeiro: UFRJ/COPPE, 2010.

XIII, 135 p.: il.; 29,7 cm.

Orientador: Rubem Pinto Mondaini

Tese (doutorado) – UFRJ/ COPPE/ Programa de Engenharia de Sistemas e Computação, 2010.

Referencias Bibliográficas: p. 121-135.

1. Reconhecimento de Padrões. 2. Redes Neurais Artificiais. 3. Biologia Computacional. I. Morais, Emerson Cordeiro. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

DEDICATÓRIA

Às mulheres de minha vida: minha esposa Adriana e minha filha Júlia.

AGRADECIMENTOS

A Deus, que iluminou meu longo caminho, mostrando as melhores soluções para tantos problemas difíceis que se apresentaram durante a realização deste trabalho.

Ao meu orientador Professor Rubem Mondaini, pela oportunidade de elaboração e acompanhamento deste trabalho, me proporcionando melhores oportunidades na vida acadêmica e profissional.

Aos membros da banca se prontificaram a participar, opinar e melhorar o trabalho.

À minha esposa Adriana Soares Morais, por todo seu amor, carinho, apoio e principalmente compreensão em tantos momentos difíceis nesta caminhada tão longa e difícil. Obrigado por tudo!

Aos meus pais, José Elias Morais e Maria do Socorro Cordeiro Morais, por toda a educação que me proporcionaram durante minha vida. A conclusão deste trabalho é mais uma vitória que eu conquistei com o apoio deles.

Aos meus colegas de Doutorado, especialmente, Reinaldo, Edgar e Sandro, pelo companheirismo, lealdade e amizade.

A todos os funcionários do Programa de Engenharia de Sistemas e Computação.

E a todos que contribuíram direta ou indiretamente para a conclusão deste trabalho.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

RECONHECIMENTO DE PADRÕES E REDES NEURAIS ARTIFICIAIS EM PREDIÇÃO DE ESTRUTRAS SECUNDÁRIAS DE PROTEÍNAS

Emerson Cordeiro Morais

Março/2010

Orientador: Rubem Pinto Mondaini

Programa: Engenharia de Sistemas e Computação

Neste trabalho propõe-se a utilização de reconhecimento de padrões e redes neurais artificiais *Perceptron* de Múltiplas Camadas (*MLP*) na predição de estruturas secundárias de proteínas. Parte-se das últimas melhorias de trabalhos anteriores de forma estanque, a saber: variação do tamanho de janelas; utilização de informações evolucionárias; utilização de júri de decisão e criação de redes neurais em cadeia, e propõe-se: reunir todas estas últimas melhorias em uma ferramenta única (*cMLP*); realizar testes com um conjunto de proteínas já testado por trabalhos anteriores; e realizar a comparação da ferramenta construída com outros classificadores já disponíveis na *web*, todos originados de trabalhos científicos, e assim, realizar um processo de otimização dos parâmetros da rede neural.

Otimizou-se a rede neural convencional com inicializações aleatórias, para torná-la uma rede neural *MLP* otimizada (*oMLP*). O processo de otimização ocorreu na utilização de parâmetros de inicialização do projeto da rede neural baseada na análise multiclasse discriminante linear e extração de subespaços com o Critério de Fisher de Pesos, inclusive na seleção de dados de entrada. Esperava-se que a rede *oMLP* demonstrasse consistentemente sua habilidade, controlando o aumento da dimensionalidade em conjuntos extensos de dados de proteínas e obtivesse melhorias significativas na maioria das medidas de desempenho, incluindo acurácia da predição da estrutura secundária e propriedades de convergência.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

PATTERN RECOGNITION AND ARTIFICIAL NEURAL NETWORKS IN THE PROTEIN SECONDARY STRUCTURE PREDICTION

Emerson Cordeiro Morais

March/2010

Advisor: Rubem Pinto Mondaini

Department: Systems and Computer Engineering

This thesis proposes to use pattern recognition and *MultiLayer Perceptron* (*MLP*) artificial neural networks in protein secondary structure prediction. We get the latest improvements of previous works, namely: variation in the size of windows; use of evolutionary information; use of decision jury and creation of neural networks chain and propose: bringing together all these improvements in a single tool (*cMLP*), to test with a set of proteins already tested by previous works and to perform the comparison of the tool built with other classifiers already available on the web, all arising from papers, and thus perform an optimization process of the neural network parameters.

The conventional neural network with random initializations was optimized, becoming an optimized *MLP* neural network (*oMLP*). The optimization process occurred in the initialization parameters of the neural network project based in the multi-class linear discriminant analysis and the weighted Fisher Criterion subspaces extraction, even in the selection input data. We hope that the *oMLP* network will be effective in the process of dimensionality reduction of the search space in extensive groups of protein data will enhance the most of the performance measures by including the accuracy of the secondary structure prediction and convergence properties.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

Capítulo I – Introdução

1.1 Considerações Iniciais

Atualmente, com a finalização do sequenciamento do genoma humano e de diversos outros organismos, deu-se início a uma nova fase para as pesquisas genéticas, denominada Proteômica. Este termo envolve a identificação de todas as proteínas expressas pelo genoma bem como a determinação de suas funções fisiológicas e patológicas. Tal conhecimento é essencial para o desenvolvimento de novos medicamentos e métodos de diagnóstico, por exemplo.

As proteínas além de desempenharem funções catalíticas como de controle e regulação do metabolismo celular, desempenham funções de defesa e transporte, dentre outras. As funções de uma proteína estão diretamente relacionadas com a sua estrutura nativa. O processo a partir do qual a proteína sai de uma conformação aleatória e alcança sua estrutura nativa é conhecido como enovelamento (*folding*). A importância de se conhecer e solucionar o problema de enovelamento de proteínas acaba refletindo-se de forma considerável sobre a biotecnologia. Em princípio, este conhecimento pode permitir o desenvolvimento de novas proteínas com aplicações abrangendo um vasto campo.

Ao se estudar o enovelamento de proteínas pretende-se descobrir quais as propriedades do polipeptídio que levam a cadeia a adotar estrutura única e estável e, também, investigar como a sequência de aminoácidos (estrutura primária) de uma proteína está relacionada com essas propriedades. Esse processo objetiva não apenas elucidar o problema do enovelamento de proteínas, mas também produzir proteínas que possuam estrutura (secundária e terciária) desejada. Para isso, métodos como Algoritmos Genéticos, Redes Neurais Artificiais e Simulações com Monte Carlo vêm sendo utilizados (BALDI, BRUNAK *et al.*, 1999) (CHANDONIA & KARPLUS, 1996) (BOHR, BOHR *et al.*, 1988) (KONO & DOI, 1994), a fim de predizer estruturas primárias (sequências) de aminoácidos que levem às estruturas secundárias ou terciárias desejadas.

Devido a sua complexidade, o problema de determinar a função de uma proteína a partir de sua estrutura primária utilizando métodos computacionais vem sendo abordado de diversas formas. A análise pode ser puramente sequencial, através da busca por padrões pré-determinados ou não, ou pode usar informações de proteínas de estruturas conhecidas, através de métodos preditivos ou comparativos.

Para a análise de sequências, métodos comparativos são tradicionalmente utilizados. A sequência sobre a qual se deseja obter mais informações é comparada com outras de função conhecida procurando alinhar resíduos idênticos ou similares, de modo a evidenciar semelhanças locais ou globais. Dependendo do grau de similaridade alcançado, a função da proteína pode ser inferida. Outra opção é efetuar a busca por ocorrências de padrões, os quais podem representar, por exemplo, sítios ativos determinados previamente, ou simplesmente subsequências que se repetem de forma exata ou aproximada (BENSON & WATERMAN, 1994) (KANNAN & MYERS, 1996) (SAGOT, 1998).

1.2 Motivação

As redes neurais artificiais foram concebidas de forma a simular em um ambiente computacional, a estrutura e a funcionalidade de redes neuronais. Suas principais vantagens são: tolerância a falhas; capacidade de aprendizagem; capacidade de auto-adaptação; e capacidade de resolver problemas práticos sem a necessidade da definição de listas de regras ou de modelos precisos.

As redes neurais artificiais oferecem melhores abordagens para problemas que requeiram: reconhecimentos de padrões; classificação de padrões; associação de padrões; identificação; resistência ao ruído; aproximação de funções e aprendizado. Pelas razões descritas acima e pela acurácia dos resultados de trabalhos anteriores (BOHR, BOHR *et al.*, 1988) (POLLASTRI, PRZYBYLSKI *et al.*, 2002) (GUIMARÃES, MELO *et al.*, 2003) (FRISHMAN & ARGOS, 1997) (ROST & SANDER, 2000), acredita-se que redes neurais artificiais podem ser perfeitamente aplicadas na predição de estruturas secundárias de proteínas.

A função de uma proteína está intrinsecamente ligada a sua conformação espacial. Métodos de obtenção de sua estrutura tridimensional, tais como difração de raio-X e ressonância magnética nuclear, são custosos e assim, muitas vezes não são passíveis de serem aplicados. Sendo assim, determinar a sequência proteica é

relativamente mais fácil do que determinar sua estrutura, fato que promove uma grande diferença entre o número de sequências e o número de estruturas conhecidas. Métodos computacionais são usados a fim de diminuir este problema baseando-se no fato de que a função proteica está diretamente relacionada com a sequência de aminoácidos que a compõe (NELSON & COX, 2000). A obtenção da estrutura terciária a partir de sua sequência é um dos principais problemas em aberto da Biologia Computacional.

Concomitantemente à predição de estrutura secundária de proteínas outro aspecto de fundamental importância deve ser considerado na análise funcional: a comparação entre estruturas. Descobrir semelhanças entre conformações protéicas ajuda na compreensão do relacionamento entre sequência, estrutura e funções. Tais similaridades podem ser evidenciadas através da busca por padrões que podem, por exemplo, caracterizar famílias de proteínas, ou seja, proteínas relacionadas funcionalmente ou estruturalmente. Neste tipo de abordagem busca-se, entre outros, por representações de padrões de enovelamento que facilitem a recuperação de informações em bancos de dados de estruturas e a determinação de relação entre as topologias de proteínas.

1.3 Objetivos

O método de predição de estruturas secundárias de proteínas utilizado neste trabalho tem o objetivo de classificar resíduos adjacentes em padrões do tipo *H* (α-hélices), *F* (folhas-β) e *C* (*coil* ou fita aleatória). A hipótese de trabalho é otimizar o processo de classificação de estruturas secundárias de proteínas utilizando o Critério de Pesos de Fisher na configuração dos parâmetros de uma rede neural.

Um ponto importante dos métodos de predição de estrutura secundária é o fato de que segmentos de resíduos consecutivos possuem preferência por certos estados de estrutura secundária. Então, o problema de predição de estrutura torna-se problema clássico de classificações de padrões, que é tratável por algoritmos de reconhecimento de padrões, por exemplo, redes neurais artificiais.

A inferência da estrutura tridimensional a partir da sequência propriamente dita também é utilizada na determinação da função de proteína. Tal problema, conhecido como *Protein Folding Problem* (Problema do Enovelamento ou Dobramento de Proteínas), foi demonstrado ser *NP*-difícil (FRAENCKEL, 1993). Devido a sua complexidade, tal procedimento é dividido em série de passos intermediários. Por

exemplo, a localização na sequência de subestruturas comuns na conformação tridimensional, tais como α-hélices, folhas-β e *coils*. Este problema é conhecido como a predição de estrutura secundária de proteínas, e é tratado neste trabalho através da abordagem que tem produzido os resultados mais satisfatórios: a predição através de redes neurais artificiais. Técnicas, algoritmos e tipos de dados utilizados nos melhores preditores disponíveis na atualidade serão explanados (POLLASTRI, PRZYBYLSKI *et al.*, 2002) (BALDI & BRUNAK, 2001) (PETERSEN, LUNDEGAARD *et al.*, 2000) (JONES, 1999) (GUIMARÃES, MELO *et al.*, 2003).

Um dos objetivos específicos é construir uma rede neural artificial *MLP* convencional (*cMLP*), isto é, com inicializações aleatórias, utilizando todas as melhorias apresentadas em trabalhos mais recentes, como: variação do tamanho de janelas, utilização de informações evolucionárias, utilização de júri de decisão e criação de redes neurais em cadeia, onde a saída da rede anterior é utilizada como entrada na arquitetura de rede posterior. A rede *cMLP* deve ser treinada e testada com um conjunto específico de proteínas já utilizado em trabalhos anteriores. Estes primeiros resultados devem ser comparados com os principais preditores apresentados nestes trabalhos.

Após este trabalho inicial, outro objetivo específico é otimizar a rede *cMLP*, para torná-la uma rede neural *MLP* otimizada (*oMLP*), e realizar novos treinamentos e testes com as mesmas proteínas. A realização desta otimização deve ocorrer na utilização de parâmetros do projeto da rede neural baseada em predições por estimativa usando ***análise multiclasse discriminante linear*** (*multi-class Linear Discriminant Analysis – LDA*) e extração de subespaços com o **Critério de Pesos de Fisher** (*weighted Fisher Criterion - wFC*). O objetivo geral deste experimento é alcançar acurácia de predição de estruturas secundárias melhor do que os resultados obtidos em trabalhos anteriores.

1.4 Organização do Trabalho

Este texto está dividido em seis capítulos, além desta introdução. Os capítulos II, III e IV realizam revisões de literatura nas áreas de Reconhecimento de Padrões e Funções Discriminantes, Redes Neurais Artificiais e Predição de Estruturas Secundárias de Proteínas, respectivamente. Estes capítulos irão fornecer fundamentos básicos para o entendimento e acompanhamento do trabalho.

O capítulo V descreve e projeta o problema que será resolvido na tese: predição de estruturas secundárias através dos diversos modelos de redes neurais artificiais, e apresenta os materiais e métodos que serão utilizados no trabalho, explicitando as arquiteturas propostas que resolveram o problema, assim como o conjunto de proteínas que foram utilizados nas fases de validação, treinamento e testes. O capítulo VI versa sobre os resultados alcançados e a comparação com resultados já publicados na literatura. Por fim, o Capítulo VII apresenta as conclusões e as possibilidades de trabalhos futuros.

Capítulo II – Reconhecimento de Padrões e Funções Discriminantes

Neste capítulo serão apresentados aspectos fundamentais de Reconhecimento de Padrões. Será abordado o problema do aumento de dimensionalidade, assim como processos para a redução do mesmo. Também será exposto teoricamente funções discriminantes, particularmente Análise de Discriminantes Lineares de Fisher (*LDA*), como um método de extração de características que será utilizado posteriormente no processo de otimização dos parâmetros de modelos de Redes Neurais Artificiais.

2.1 Introdução

A abrangência do reconhecimento de padrões está na sua efetiva realização para as diversas áreas de pesquisas. A área abrange desde a detecção de padrões à escolha mais simples entre dois objetos, como a complexa realização da aprendizagem.

Segundo De Campos (2001), o reconhecimento de padrões está relacionado aos aspectos associados a problemas específicos, como por exemplo, redução de dimensionalidade, extração de características e escolha de um classificador adequado.

Existem muitas definições e muitas abordagens sobre Reconhecimento de Padrões (*RP*). Duda, Hart & Stork (2000), caracterizaram *RP* como: "campo que consiste no reconhecimento de regularidades significativas em meios ruidosos e complexos." Ainda, Bezdek e Pal (1992) definem *RP* como: "a busca por estruturas em dados".

Segundo Pao (1989), a área de reconhecimento de padrões é importante devido às ocorrências na vida humana tomarem forma de padrões. A formação da linguagem, o modo de falar, o desenho das figuras, o entendimento das imagens, tudo envolve padrões. *RP* é uma tarefa complexa, onde se busca, sempre, avaliar as situações em termos dos padrões das circunstâncias que as constituem, descobrir relações existentes no meio, para melhor entendê-las e adaptar-se.

Basicamente, o reconhecimento de padrões é a área de pesquisa que tem por objetivo a classificação de objetos (padrões) em número de categorias ou classes

(THEODORIDIS & KOUTROUMBAS, 1999). Assim, dado conjunto de c classes, ω_1, ω_2, ..., ω_c, e padrão desconhecido x, reconhecedor de padrões é um sistema que, auxiliado por pré-processamentos, extração e seleção de características, associa x ao rótulo i de uma classe ω_i. No caso de classificação de proteínas, a sequência de aminoácidos é o objeto (ou padrão x) e as classes são suas estruturas secundárias (ω_i).

Segundo Jain *et al.* (2000), nos últimos 50 anos de pesquisa, foram obtidos avanços que possibilitaram a evolução da pesquisa em aplicações altamente complexas. Os autores destacam os seguintes exemplos de aplicações atuais que requerem técnicas eficientes e robustas de reconhecimento de padrões:

a) Bioinformática: análise de sequências do genoma; aplicações e tecnologia de *microarrays*;
b) Diagnóstico médico;
c) Mineração de dados (*Data Mining*): a busca por padrões significativos em espaços multi-dimensionais, normalmente obtidos de grandes bases de dados e *Data Warehouses*;
d) Classificação de documentos da Internet;
e) Análise de imagens de documentos para reconhecimento de caracteres (*Optical Character Rocognition - OCR*);
f) Inspeção visual para automação industrial;
g) Busca e classificação em base de dados multimídia;
h) Reconhecimento biométrico, incluindo faces, íris ou impressões digitais;
i) Sensoriamento remoto por imagens multiespectrais;
j) Reconhecimento de fala.

Um ponto em comum a essas aplicações é que usualmente as características disponíveis nos padrões de entrada, tipicamente milhares, não são diretamente utilizadas. Normalmente utilizam-se características extraídas dos padrões de entrada otimizadas através de procedimentos guiados pelos dados, como *LDA* (vide seção 2.6.4).

O projeto de sistemas de reconhecimento de padrões essencialmente envolve três aspectos: aquisição de dados e pré-processamento, representação dos dados e tomada de decisões. Geralmente o desafio encontra-se na escolha de técnicas para efetuar esses três aspectos.

Um problema de reconhecimento de padrões bem definido e restrito permite uma representação compacta dos padrões e uma estratégia de decisão simples. Seja

$d_w(\omega_i)$ uma medida de separabilidade global entre os padrões pertencentes a uma classe ω_i (por exemplo, a média das variâncias em todas as características dos padrões de ω_i). Seja $d_b(\Omega)$ uma medida de separabilidade global entre as classes do conjunto de classes Ω (por exemplo, a média das distâncias entre as médias de todas as classes de Ω e a média global). Um problema de reconhecimento de padrões bem definido e restrito é aquele que, em seu espaço de características, possui distribuições de padrões com pequenas variações intra-classes e grande variação inter-classes, ou seja, pequenos valores de $d_w(\omega_i)$ e um grande valor de $d_b(\Omega)$ (THEODORIDIS & KOUTROUMBAS, 1999).

Geralmente, em dados reais, os padrões a serem reconhecidos não possuem essas peculiaridades. Nesse fato reside a importância de algoritmos de extração e seleção de características, pois eles reduzem a dimensionalidade dando prioridade para uma base do espaço de características que não perde o poder de discriminação dos padrões. A seguir serão traçados detalhes a respeito dos métodos de reconhecimento estatístico de padrão.

2.2 Abordagem Estatística de Reconhecimento de Padrões

Existem várias abordagens diferentes para se efetuar reconhecimento de padrões. Dentre elas, podemos destacar:

- Casamento ou associação (*template matching*) (GONZALEZ & WOODS, 1992) (FERRIS *et al.*, 2000) (THEODORIDIS & KOUTROUMBAS, 1999);
- Abordagem sintática (por exemplo: *Hidden Markov Models*) (THEODORIDIS & KOUTROUMBAS, 1999) (MORIMOTO *et al.*, 1996);
- Redes neurais (THEODORIDIS & KOUTROUMBAS, 1999);
- Lógica nebulosa (BLOCH, 1999) (DUBOIS *et al.*, 1997) (BOVENTI-JR & COSTA, 2000);
- Morfologia matemática com aprendizado computacional (BARRERA *et al.*, 2000);
- Estatística.

É importante ressalvar que essa separação entre as abordagens possui apenas fins didáticos, pois, apesar de possuírem aparentemente princípios diferentes, a maioria dos modelos de redes neurais populares são equivalentes ou similares a métodos clássicos de reconhecimento estatístico de padrões. Entretanto, algumas redes neurais

podem oferecer certas vantagens, como abordagens unificadas para extração de características, seleção de características e classificação, e procedimentos flexíveis para encontrar boas soluções não lineares (JAIN *et al.*, 2000).

Basicamente, um sistema de reconhecimento estatístico de padrões pode ser composto pelas seguintes partes (DUDA, HART & STORK, 2000) (JAIN *et al.*, 1999) (vide Figura 2.1): sistema de aquisição de dados; sistema de pré-processamento, para eliminar ruídos ou distorções; extrator de características (ou atributos), que cria um vetor de características com dados extraídos dos objetos adquiridos, reduzindo os dados a atributos, propriedades ou características; seletor de características, que analisa o conjunto de características e elimina as mais redundantes; e classificador, que analisa um padrão obtido e toma uma certa decisão.

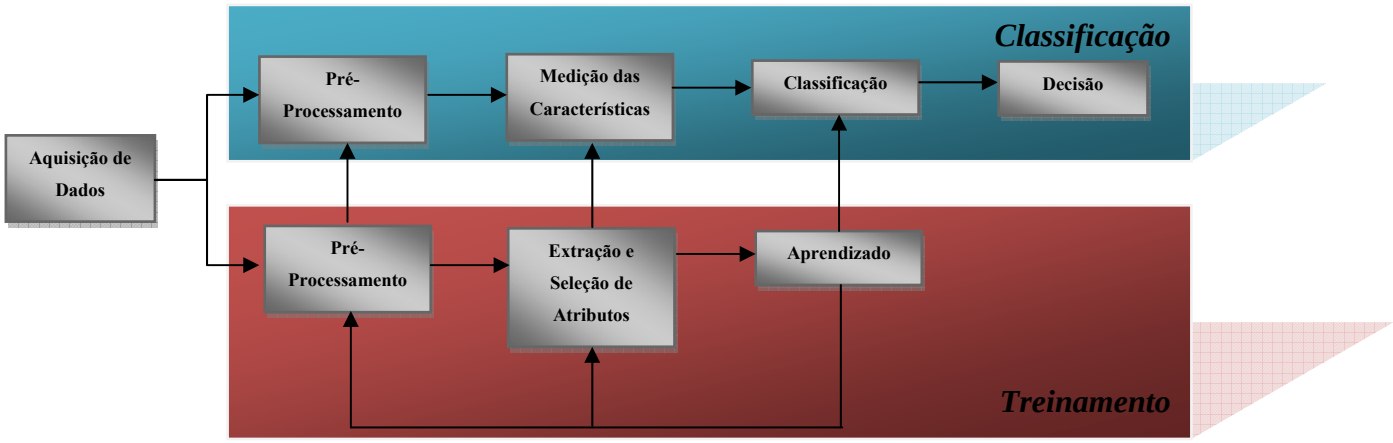

Figura 2.1: Sistema Genérico de Reconhecimento de Padrões

O classificador toma decisões baseando-se no aprendizado realizado a partir de um conjunto de treinamento, o qual contém exemplos de padrões de todas as classes existentes no sistema. Conforme será detalhado posteriormente, em reconhecimento estatístico de padrões, a classificação é realizada utilizando estimativas de distribuições probabilísticas, por isso o nome dessa abordagem. O reconhecedor de padrões é avaliado através de um conjunto de testes, preferencialmente composto por padrões de todas as classes, mas que não estejam no conjunto de treinamento. Além do classificador, o pré-processamento, o extrator e o seletor de características podem ser dependentes dos dados de treinamento.

No caso de sistemas estatísticos, quando o problema abordado for muito complexo, torna-se essencial o uso de extração e seleção de características. Exemplos de problemas complexos são aqueles em que há muitas classes ou quando a dimensão dos padrões no formato em que são adquiridos for muito alta.

Em uma abordagem de *RP*, cada padrão é representado em termos de N características (*features*) ou atributos. Um padrão é representado por vetor de características $\mathbf{x} = [x_1, x_2, \cdots, x_N]^t$, modelado como vetor aleatório, em que cada x_j ($1 \leq j \leq N$) é uma característica (THEODORIDIS & KOUTROUMBAS, 1999). Cada padrão medido x_i é uma instância de **x**. O espaço formado pelos vetores de características é chamado de espaço de características, o qual possui dimensão N.

Uma classe ω_i (a i-ésima classe de um conjunto de classes Ω, de c classes) é um conjunto que contém padrões os quais possuem alguma relação ou peculiaridade em comum. Em exemplo simples de biometria, podemos ter espaço de características **x** em que x_1 representa altura, x_2 representa peso e x_3 representa o tamanho dos pés. Nesse mesmo espaço de três dimensões, cada instância de **x** representa as medições tomadas de uma pessoa em um determinado instante. Cada classe representa uma família de pessoas. Nesse caso, o problema de classificação define-se por: dada uma pessoa desconhecida, extrair suas características para obter seu vetor de características **x** e determinar a qual família provavelmente essa pessoa pertence.

Os padrões são tratados como vetores aleatórios, pois um padrão desconhecido pode ser o representante de uma classe conhecida que sofreu alterações aleatórias proporcionadas por ruídos oriundos do método de aquisição (sensores), da influência de outros fatores externos ou mesmo dos mecanismos de extração de características intrínsecos ao sistema de reconhecimento.

2.3 Métodos de Classificação

Dado um padrão desconhecido **x**, pertencente ao conjunto padrões de teste X em um espaço de características, e o conjunto Ω de todas as classes existentes, um classificador é uma função $Y: X \rightarrow \Omega$, tal que $Y(\mathbf{x}) = \omega_i$, em que ω_i é a i-ésima classe de Ω. Assim, um classificador é uma função que possui como entrada padrões desconhecidos e, como saída, rótulos que identificam a que classe tais padrões provavelmente pertencem (essa definição é válida para todos os classificadores, não só para os estatísticos). Portanto, classificadores são elementos que, de fato, realizam o reconhecimento de padrões. Todos os classificadores devem ser treinados utilizando um conjunto de amostras.

Esse treinamento é utilizado pelo algoritmo do classificador para determinar as *fronteiras de decisão* do espaço de características. Fronteiras de decisão são superfícies

multidimensionais no espaço de características F, que particionam F em c regiões para um problema com c classes, cada região correspondendo a uma classe. Se as regiões S_i e S_j são contíguas, são separadas por uma superfície de decisão. Assim, tem-se $F = \bigcup_{i=1}^{c} S_i$. A regra de decisão faz com que um padrão desconhecido que se encontra na região S_i do espaço de características seja rotulado como um padrão da classe ω_i, ou seja, $Y(\mathbf{x}) = \omega_i$.

Dessa forma, essencialmente, o que difere um classificador de outro é a forma como esse cria as fronteiras de decisão a partir dos exemplos de treinamento. Os exemplos de treinamento de cada classe podem ser pré-especificados (aprendizado supervisionado) ou aprendidos com base nos exemplos (aprendizado não-supervisionado), como será apresentado na seção 3.3. No caso de classificação de estruturas de proteínas, normalmente é realizado aprendizado supervisionado, isto é, as proteínas de treinamento possuem um rótulo que identifica quais estruturas secundárias são encontradas nela e devido a que conjunto de aminoácidos.

Apesar da existência de vários algoritmos diferentes para determinar fronteiras de decisão (métodos de classificação), pode-se dizer que todos têm em comum os seguintes objetivos:

1. Minimizar o erro de classificação; e
2. Permitir que a classificação seja eficiente computacionalmente, isto é, leve o menor tempo possível para realizar o processamento.

Porém, a importância de cada um desses objetivos varia de classificador para classificador. Obviamente, o ideal é que um classificador seja rápido e apurado, mas, em problemas complexos, isto quase nunca ocorre. A seguir, há maiores detalhes de alguns classificadores que de alguma forma foram utilizados neste trabalho.

2.3.1 Classificador Bayesiano

A fim de possibilitar a formalização dos classificadores utilizados neste trabalho, inicialmente serão descritos alguns pontos da teoria da decisão e de um classificador Bayesiano. Antes de definir um classificador Bayesiano é necessário definir os conceitos a seguir.

Probabilidade a priori de uma classe

Um dado vetor **x** pode ser associado a uma classe *i* de *c* classes ω_1, ω_2, ω_3,⋯, ω_c, com uma probabilidade P_i, chamada de probabilidade a priori da classe *i*, com $\sum_{i=1}^{c} P_i = 1$.

Probabilidade a posteriori

Dado um padrão **x** com classificação desconhecida, a probabilidade de **x** ser da classe ω_j é $P(\omega_j|\mathbf{x})$, que é a probabilidade a posteriori da classe ω_j. Pela regra de Bayes temos:

$$P(\omega_j \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \omega_j) \cdot P_j}{p(\mathbf{x})} \tag{2.1}$$

com

$$p(\mathbf{x}) = \sum_{j=1}^{c} p(\mathbf{x} \mid \omega_j) \cdot P_j \tag{2.2}$$

Taxa de probabilidade de erro

A probabilidade de erro de classificação ao se associar um dado vetor de atributos **x** à classe ω_i é definida por:

$$e_i(\mathbf{x}) = 1 - P(\omega_i \mid \mathbf{x}),\ i = 1, \ldots, c \tag{2.3}$$

Essa é uma definição geral, sendo válida para regra de decisão arbitrária. O valor esperado dessa probabilidade sobre todos os vetores **x** pertencentes à região S_i de decisão para a classe ω_i é a probabilidade de classificação errada em ω_i, denotada ξ_i. Essa é a probabilidade de cometer-se um erro ao atribuir um vetor **x** à classe ω_i:

$$\xi_i = \int_{S_i} e_i(\mathbf{x}) \cdot p(\mathbf{x}) d\mathbf{x} = \int_{S_i} \left[1 - P(\omega_i \mid \mathbf{x})\right] \cdot p(\mathbf{x}) d\mathbf{x} \tag{2.4}$$

em que S_i é a região de aceitação associada à classe ω_i. Como a classificação de um vetor **x** só pode ocorrer nas classes mutuamente exclusivas ω_1,⋯, ω_c, segue que a probabilidade global de erro, ou taxa de erro, é a soma das probabilidades de erro ξ_i em cada classe:

$$\xi = \sum_{i=1}^{c} \xi_i = \sum_{i=1}^{c} \int_{S_i} \left[1 - P(\omega_i \mid \mathbf{x})\right] \cdot p(\mathbf{x}) d\mathbf{x} \tag{2.5}$$

A expressão entre colchetes é a probabilidade condicional de erro $e_i(\mathbf{x})$; ξ_i é a média dessa probabilidade para todo x ϵ S_i e, portanto, é a probabilidade de classificação errada em ω_i.

Infelizmente, na maioria dos casos, o cálculo da probabilidade de erro é extremamente difícil e raramente consegue-se chegar a uma expressão explícita. Na prática, a taxa de erro é geralmente estimada a partir de um conjunto de teste (conjunto de amostras de vetores com classificação conhecida).

Classificador para mínima taxa de erro

A partir da formalização da taxa ou probabilidade de erro, pode-se descrever um classificador que minimiza esse quantificador de desempenho. Inicialmente, é necessário mostrar definições duais às das equações 2.3, 2.4 e 2.5. A probabilidade de acerto ao se classificar um dado x em ω_i é:

$$a_i(\mathbf{x}) = P(\omega_i \mid \mathbf{x}),\ i = 1, \ldots, c \tag{2.6}$$

A probabilidade de acerto ao se atribuir um vetor à classe ω_i é:

$$A_i = \int_{S_i} a_i(\mathbf{x}) \cdot p(\mathbf{x}) d\mathbf{x} = \int_{S_i} P(\omega_i \mid \mathbf{x}) \cdot p(\mathbf{x}) d\mathbf{x} \tag{2.7}$$

A probabilidade de classificação correta ou probabilidade de acerto ou taxa de acerto é:

$$A = \sum_{i=1}^{c} A_i = \sum_{i=1}^{c} \int_{S_i} P(\omega_i \mid \mathbf{x}) \cdot p(\mathbf{x}) d\mathbf{x} \tag{2.8}$$

Obviamente, a mínima taxa de erro é obtida quando a taxa de acerto é máxima

$$\min \xi \Leftrightarrow \max_{S_i} \sum_{i=1}^{c} \int_{S_i} P(\omega_i \mid \mathbf{x}) \cdot p(\mathbf{x}) d\mathbf{x} \tag{2.9}$$

A máxima taxa de acerto é obtida quando cada S_i é escolhido como o domínio onde $P(\omega_i|\mathbf{x}) \geq P(\omega_j|\mathbf{x}),\ \forall j$.

Assim, o classificador bayesiano de mínima taxa de erro pode ser definido como:

$$\Upsilon(\mathbf{x}) = \omega_i \quad \text{se } \mathbf{x} \in S_i \tag{2.10}$$

com

$$S_i = \left\{ \forall \mathbf{x} \in F \ \text{ tal que } P(\omega_i \mid \mathbf{x}) \geq P(\omega_j \mid \mathbf{x}), j = 1, \ldots, c \right\} \tag{2.11}$$

Ou simplesmente,

$$\Upsilon(\mathbf{x}) = \omega_i \ \text{ se } \ P(\omega_i \mid \mathbf{x}) \geq P(\omega_j \mid \mathbf{x}), j = 1, \ldots, c \tag{2.12}$$

Após essa descrição do classificador de Bayes de mínima taxa de erro, a seguinte questão ingênua pode surgir: se o classificador Bayesiano é um classificador ótimo, então por que outros classificadores são utilizados? O motivo é que o classificador de Bayes só pode ser executado se a probabilidade a priori P_i e as probabilidades $p(\mathbf{x}|\omega_i)$ forem conhecidas.. Em problemas práticos, na fase de treinamento são utilizados métodos de estimação dessas probabilidades. Entretanto, quando a distribuição das classes possui formas "complicadas" e descontínuas, o preço computacional desses métodos torna-se muito alto quando se deseja obter uma representação precisa dessas probabilidades (DE CAMPOS, 2001).

Uma abordagem para se resolver esse problema é considerar um modelo de distribuição de probabilidade para $p(\mathbf{x}|\omega_i)$. A estimativa de distribuição mais bem conhecida e, provavelmente, uma das mais simples, é a de distribuição normal ou gaussiana, dada por:

$$p(\mathbf{x} \mid \omega_i) = \frac{1}{(2\pi)^{N/2} \cdot \sqrt{\det(\Sigma_i)}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu_i)^t \cdot \sum_i^{-1} \cdot (\mathbf{x} - \mu_i) \right), i = 1, \ldots, c \qquad \textbf{(2.13)}$$

em que $\mu_i = E[\mathbf{x}]$ é o valor esperado (tomado pela média) da classe ω_i, e Σ_i é a matriz de covariância N x N definida por:

$$\sum_i = E\left[(\mathbf{x} - \mu_i) \cdot (\mathbf{x} - \mu_i)^t\right] \qquad \textbf{(2.14)}$$

det (Σ_i) denota o determinante de Σ_i e $E[\cdot]$ a média (ou esperança) de uma variável aleatória. É comum o uso do símbolo $N(\mu,\Sigma)$ para denotar a função de densidade probabilística Gaussiana. A partir dessas definições e das anteriores, constrói-se o classificador Bayesiano para distribuições normais.

2.3.2 Regra dos *K*-vizinhos mais Próximos

A regra de classificação dos *K*–vizinhos mais próximos (*KNN – K–Nearest Neighbours*) é um método de classificação que não possui processamento na fase de treinamento, pois não é necessário estimar as distribuições de probabilidades das classes. Entretanto, é necessário um grande número de padrões de treinamento (padrões cuja classe é conhecida a priori), pois se pode dizer que as tarefas de estimativa e de classificação são fundidas em uma única tarefa. O classificador *KNN* é um classificador sub-ótimo que cria fronteiras de decisão complexas.

Dado um padrão de teste (desconhecido) **x**, sua classificação é realizada da seguinte maneira:

- Inicialmente, calcula-se a distância entre **x** e todos os padrões de treinamento;
- Verifica-se a quais classes pertencem os *K* padrões mais próximos;
- A classificação é feita associando-se o padrão de teste à classe que for mais frequente entre os *K* padrões mais próximos de **x**.

Há duas distâncias que normalmente são adotadas para implementar esse classificador:

Distância Euclidiana

A distância Euclidiana entre dois vetores ($\mathbf{x_i}$ e $\mathbf{x_j}$) e definida por:

$$d_{\varepsilon}(\mathbf{x}_i, \mathbf{x}_j) = \left\| \mathbf{x}_i - \mathbf{x}_j \right\| = \sqrt{\left(\mathbf{x}_i - \mathbf{x}_j\right)^t \cdot \left(\mathbf{x}_i - \mathbf{x}_j\right)} \qquad \textbf{(2.15)}$$

Distância de Mahalanobis

A distância de Mahalanobis entre um padrão **x** e o protótipo σ de uma classe é definida por:

$$d_{\mathrm{M}}(\mathbf{x}, \sigma) = \sqrt{\left(\mathbf{x} - \sigma\right)^t \cdot \Sigma^{-1} \cdot \left(\mathbf{x} - \sigma\right)} \qquad \textbf{(2.16)}$$

em que Σ é a matriz de covariância dos padrões da classe de σ, determinada a seguir.

Dados |T| padrões de treinamento, $\mathbf{x}_1$, $\mathbf{x}_2$, ..., $\mathbf{x}_{|T|}$, a matriz de covariância destes padrões de treinamento **X**. É definida como uma matriz em que cada coluna possui um padrão de treinamento:

$$\mathbf{X} = \left[\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_{|T|}\right] \qquad \textbf{(2.17)}$$

Dada uma matriz de padrões X, a matriz de covariância Σ_X pode ser obtida a partir da equação 2.14:

$$\sum_{\mathbf{x}} = (\mathbf{X} - \mu) \cdot (\mathbf{X} - \mu)^t \qquad \textbf{(2.18)}$$

Em que μ é a matriz **N** x |T|, e todas as colunas contêm o valor esperado dos padrões de **X**:

$$\mu_{l,j} = \frac{1}{|T|} \cdot \sum_{j=1}^{|T|} \mathbf{X}_{l,j} \qquad \textbf{(2.19)}$$

para l = 1, 2, 3, ..., N e i = 1, 2, 3, ..., |T|.

Normalmente, a regra de classificação por vizinho mais próximo acarreta numa taxa de erro maior do que a da regra de decisão de Bayes. Porém existe um teorema que diz que, supondo-se que haja infinitos padrões de treinamento, a taxa de erro desse

classificador não ultrapassa (sendo em geral menor que) o dobro da taxa de erro com o classificador de Bayes (ver demonstração em Theodoridis & Koutroumbas (1999)).

O classificador *KNN* pode ser descrito formalmente utilizando o classificador de Bayes com mínima taxa de erro. A desigualdade contida na equação 2.12 equivale a $P(\omega_i|\mathbf{x}) \geq P(\omega_j|\mathbf{x})$, contando que $p(\mathbf{x}) \neq 0$. Para estimar P_i a partir dos dados, basta tomar $|T_i| / |T|$, em que $|T|$ é o número total de amostras e $|T_i|$ é o número de amostras na classe ω_i. Para se estimar $p(\mathbf{x}|\omega_i)$, pode-se tomar um volume $B_\mathbf{x}$, centrado em $\mathbf{x}$ e contar-se quantas amostras há em seu interior. Dessa forma:

$$decidir\ \omega_i \ \text{ se } \frac{|T_i|}{|T|} \cdot \frac{K_i}{|T_i| \cdot B_\mathbf{x}} \geq \frac{|T_j|}{|T|} \cdot \frac{K_j}{|T_j| \cdot B_\mathbf{x}} \quad j = 1, ..., c \qquad \textbf{(2.20)}$$

Em que se supõe que volume $B_\mathbf{x}$ abarca K_i amostras indistintamente das classes envolvidas, com $K = \sum_{i=1}^{c} K_i$. Simplificando,

$$decidir\ \omega_i \ \text{ se } \frac{K_i}{|T| \cdot B_\mathbf{x}} \geq \frac{K_j}{|T| \cdot B_\mathbf{x}} \quad j = 1, ..., c \qquad \textbf{(2.21)}$$

A principal vantagem desse método é que ele cria uma superfície de decisão que se adapta à forma de distribuição dos dados de treinamento de maneira detalhada, possibilitando a obtenção de boas taxas de acerto quando o conjunto de treinamento é grande ou representativo. O objetivo de se utilizar $K > 1$ é reduzir a ocorrência de erros causados por ruídos nos padrões de treinamento. Por exemplo, um padrão de treinamento $\mathbf{x}_r$ da classe ω_i que se encontra em uma região do espaço de características povoada por padrões de treinamento da classe ω_j devido à ação de ruídos não prejudicará o desempenho do classificador, pois a verificação de seus vizinhos fará com que um padrão de teste que se localize próximo a $\mathbf{x}_r$ seja classificado como um padrão da classe ω_j. Porém, o uso de valores grandes em K pode reduzir a qualidade dos resultados de classificação quando a distribuição das classes possui muitas sobreposições (THEODORIDIS & KOUTROUMBAS, 1999).

A principal desvantagem dos classificadores *KNN* está em sua complexidade na fase de testes. Isso se deve ao fato de que, caso seja feita uma busca em "força-bruta" (sem ordenação) pelos vizinhos mais próximos, para cada padrão de teste é necessário realizar $K \cdot |T|$ medições de distância, ou seja, a quantidade de operações necessárias é da ordem de $K \cdot O(|T|)$.

2.4 Problemas de Generalização

Nesta seção, são discutidos os problemas de generalização de classificadores. Tais problemas são muito relevantes no projeto de sistemas de reconhecimento estatístico de padrões, que também podem ser comuns a sistemas não estatísticos, como redes neurais.

Não importando qual o classificador utilizado, em problemas práticos, ele deve ser treinado usando exemplos de treinamento para estimar a distribuição das classes. Como resultado, o desempenho do classificador depende tanto do número de exemplos de treinamento como dos valores específicos das instâncias, ou seja, da qualidade desses exemplos. Ao mesmo tempo, o objetivo do projeto de um sistema de reconhecimento é classificar futuros exemplos de teste mesmo que esses não sejam os mesmos que os de treinamento.

Porém, a otimização de classificador para maximizar seu desempenho no conjunto de treinamento nem sempre produz bom resultado para o conjunto de testes. A habilidade de *generalização* de classificadores refere-se a seu desempenho ao classificar padrões de teste que não foram utilizados durante o treinamento.

Os problemas de generalização ocorrem quando o classificador utiliza mais informações (*features*) em seus padrões de treinamento que as necessárias. Basicamente, há três problemas oriundos da redução na capacidade de generalização de um classificador (JAIN *et al.*, 2000):

- Sobre-ajuste (*overfitting*), relacionado com o número de parâmetros livres do classificador;
- Sobre-treinamento (*overtraining*), relacionado com o número de iterações de treinamento;
- Problema da dimensionalidade (*curse of dimensionality*), relacionado com a dimensão do espaço de características.

Assim, o desempenho de um classificador depende da relação entre sua complexidade e a qualidade do conjunto de treinamento (o quanto ele representa a distribuição dos dados). A taxa de erro dos classificadores apresenta um comportamento de curva em *U* com a variação dos fatores relacionados com esses problemas. A seguir, encontram-se mais detalhes sobre o problema da dimensionalidade, pois afeta todos os sistemas de reconhecimento de padrão estatístico e também por causa da sua relação com seleção de características.

O Problema da Dimensionalidade

O termo dimensionalidade é atribuído ao número de características de uma representação de padrões, ou seja, a dimensão do espaço de características (N). O problema da dimensionalidade trata-se do seguinte fenômeno: o número de elementos de treinamento requeridos para que um classificador tenha um bom desempenho é uma função monotonicamente crescente da dimensão do espaço de características.

Em alguns casos (mas não necessariamente em todos), pode-se mostrar que essa função é exponencial (JAIN *et al.*, 2000). Um exemplo é o da técnica de particionamento do espaço de características para classificação baseada em árvores de decisão. Nessa técnica, cada reta suporte dos vetores da base do espaço de características é segmentado em intervalos regulares. A interseção entre esses intervalos forma células no espaço. O reconhecimento de padrões é feito através da associação de uma classe a cada célula, de acordo com a classe majoritária nas células. Esse é um exemplo de sistema de classificação em que é bastante intuitivo verificar que, para que não haja células com classificação indefinida, é necessário que o número de elementos de treinamento seja uma função exponencial da dimensão do espaço de características. Isso ocorre devido ao fato de que, em reconhecimento estatístico de padrões, o volume do espaço de característica cresce exponencialmente com o aumento da dimensionalidade (PERLOVSKY, 1998). Esse fenômeno é bem conhecido pela comunidade de reconhecimento de padrões.

Quando é utilizado um classificador Bayesiano, nos casos em que o número de elementos de treinamento é arbitrariamente grande ou a função densidade de probabilidade das classes ($p(\mathbf{x}|\omega_i)$, $i = 1, ..., c$) for completamente conhecida, a probabilidade de erro de classificação de uma regra de decisão não aumenta com o número de características consideradas. Na prática, podemos observar que para um conjunto de treinamento finito, observa-se que a adição de características pode prejudicar o desempenho de um classificador (se não forem adicionados exemplos de treinamento). Isso ocorre quando o número de exemplos de treinamento não é grande o suficiente em relação ao número de características. Esse fenômeno, chamado fenômeno do pico (*peaking phenomena*), é uma consequência do problema da dimensionalidade, tendo também sido amplamente estudado.

Apesar de ser teoricamente clara a relação entre a dimensionalidade e o tamanho do conjunto de treinamento ($|T| \leftrightarrow O(e^N)$), sendo N a dimensão do espaço de características e T o conjunto de exemplos, há outros fatores que, quando considerados,

comprometem a exatidão dessa relação, tais como a complexidade do classificador e o número de classes. Segundo Jain *et al.* (2000), resultados empíricos fazem com que, geralmente, seja aceita a seguinte relação: $|T_i| \leftrightarrow 10 \cdot N$, $i = 1, \cdots, c$, sendo $|T_i|$ o número de exemplos de treinamento da classe i. Ou seja, no mínimo deve-se utilizar um número de exemplos de treinamento por classe dez vezes maior que a dimensionalidade.

Assim, para obter-se o desempenho máximo de um classificador, é necessário investigar qual é a dimensionalidade ideal para um determinado problema de reconhecimento de padrões. Para isto pode ser aplicada a estratégia simples de tentativa e erro em relação à dimensionalidade, usando um método de redução de dimensionalidade (incluindo extração e seleção de características) até que o ponto de máximo desempenho de um classificador seja atingido. Nessa estratégia, são realizados testes de redução de dimensionalidade para a obtenção de subespaços de características de vários tamanhos diferentes, até que seja obtida a dimensionalidade que minimiza o erro de classificação.

2.5 Redução de Dimensionalidade e Extração de Características

Existem três principais razões para que a dimensionalidade seja a menor possível. As duas primeiras são: custo de medição e precisão do classificador. A terceira versa sobre quando o espaço de características contém somente as características mais notáveis e o classificador será mais rápido e ocupará menos memória (JAIN *et al.*, 2000).

Para efetuar redução de dimensionalidade, existem basicamente duas abordagens: extração de características e seleção de características. Em linhas gerais, os algoritmos de extração criam novas características a partir de transformações ou combinações do conjunto de características original. Já os algoritmos de seleção, como o próprio nome diz, selecionam segundo determinado critério, o melhor subconjunto do conjunto de características original.

Frequentemente, a extração de características precede a seleção, de forma que, inicialmente, é feita a extração de características a partir dos dados de entrada, seguido por um algoritmo de seleção de características que elimina os atributos mais irrelevantes segundo um determinado critério, reduzindo a dimensionalidade.

A escolha entre seleção e extração de características depende do domínio de aplicação e do conjunto específico de dados de treinamento disponíveis. Em geral, a

seleção de características reduz o custo de medição de dados, e as características selecionadas mantêm sua interpretação física original, mantendo as propriedades que possuíam quando foram identificadas. Já as características transformadas geradas por extração podem fornecer uma habilidade de discriminação melhor que o melhor subconjunto das características originais.

É importante lembrar que, se a redução de dimensionalidade for excessiva, o classificador pode ter seu poder de discriminação reduzido (vide o problema da dimensionalidade na seção 2.4). Por isso, é importante analisar a variação do comportamento do classificador com o número de características, de forma que seja possível estimar a dimensionalidade ideal para determinado classificador e conjunto de dados. Para direcionar o texto de revisão da literatura em relação a este trabalho, a seguir, encontram-se maiores detalhes sobre a extração de atributos, especificamente a técnica utilizada na otimização das Redes Neurais, a Análise Discriminante Linear (*LDA*).

Um método de extração de características cria um novo espaço a partir de transformações ou combinações das características do espaço original. Formalmente, dado um espaço de características I de dimensão N, um método de extração de características H é uma função $H : I \rightarrow F$, em que F possui dimensão m. Assim, dado um padrão $\mathbf{x} \in I$, temos:

$$H(x) = y \tag{2.22}$$

tal que y ($y \in F$) é a nova representação do padrão no espaço F.

Normalmente, $m \ll N$, mas nem sempre a redução de dimensionalidade é promovida diretamente pelos métodos de extração de características. Há métodos lineares e não lineares de extração de características. Os processos lineares de extração de características podem ser definidos como uma simples mudança de base do espaço vetorial de características da seguinte forma:

$$y = H^t \cdot x \tag{2.23}$$

em que H é uma matriz mudança de base que leva elementos da base I para a base F.

Dentre os extratores de características lineares, podemos citar a transformada de Fourier, a análise de componentes principais (*PCA*), a análise de discriminantes lineares (Discriminantes de Fisher), o algoritmo perceptron, e outras projeções lineares em geral. Em relação aos extratores não lineares, pode-se citar as redes neurais multicamadas e de funções base radiais e máquinas de vetores de suporte.

A seguir, está descrito o método de extração de características que foi utilizado no desenvolvimento desta pesquisa, o método de *funções discriminantes*.Esta escolha foi realizada porque apesar de ser um dos métodos mais simples, é um dos mais adequados para a otimização dos parâmetros da rede neural utilizada para classificação de estruturas secundárias de proteínas.

2.6 Funções Discriminantes

Segundo Bishop (2006), um discriminante é uma função que toma um vetor de entrada $\mathbf{x}$ e o associa a uma das K classes, denotada por C_K. Neste trabalho, restringiremos a atenção aos discriminantes lineares chamados assim por apresentarem hiperplanos como superfícies de decisão. Para simplificar o entendimento, iremos considerar primeiramente o caso de duas classes e após, estender o estudo para casos em que $K > 2$.

2.6.1 Função Discriminante para Duas Classes

A mais simples representação de uma função discriminante linear é obtida através da função linear de um vetor de entrada tal que:

$$y(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0 \qquad \textbf{(2.24)}$$

Onde $\mathbf{w}$ é chamado *vetor de pesos*, e w_0 é a tendência ou inclinação (*bias*). O negativo da inclinação é chamado de limiar (*threshold*). Um vetor de entrada $\mathbf{x}$ é associado à classe C_1 se $y(\mathbf{x}) \geq 0$ e à classe C_2, caso contrário. O limite de decisão correspondente é, então, definido por $y(\mathbf{x}) = 0$, que corresponde a um hiperplano $(D - 1)$-dimensional com um espaço de entrada D-dimensional (neste exemplo, $D = 2$).

Considere dois pontos $\mathbf{x}_A$ e $\mathbf{x}_B$, ambos arbitrários na superfície de decisão. Devido $y(\mathbf{x}_A) = y(\mathbf{x}_B) = 0$, temos $\mathbf{w}^{\mathrm{T}}\cdot(\mathbf{x}_A - \mathbf{x}_B) = 0$ e consequentemente o vetor $\mathbf{w}$ é ortogonal a cada vetor hipotético na superfície de decisão, então $\mathbf{w}$ determina a orientação da superfície de decisão. Similarmente, se $\mathbf{x}$ é um ponto na superfície de decisão, então $y(\mathbf{x}) = 0$, e a distância normal a partir da origem até a superfície de decisão é dada por:

$$\frac{\mathbf{w}^{\mathrm{T}}\mathbf{x}}{\|\mathbf{w}\|} = -\frac{w_0}{\|\mathbf{w}\|} \qquad \textbf{(2.25)}$$

Assim vemos que o parâmetro w_0 determina a localização da superfície de decisão. Estas propriedades são ilustradas para o caso de $D = 2$ na Figura 2.2.

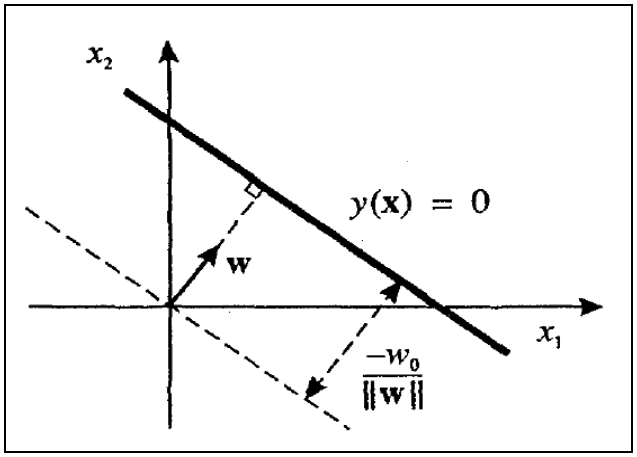

Figura 2.2: Geometria da função linear discriminante em duas dimensões

Na Figura 2.2, a superfície de decisão (linha mais forte) é perpendicular a **w**, e sua distância a partir da origem é controlada pelo parâmetro w_0. Também, a distância ortogonal associada do ponto x, a partir da superfície de decisão, é dada por $y(\mathbf{x})/\|\mathbf{w}\|$. Além disso, nota-se que o valor de $y(\mathbf{x})$ fornece uma medida associada da distância perpendicular r do ponto **x** a partir da superfície de decisão. Para ver isto, deve-se considerar um ponto arbitrário **x** e $\mathrm{x}_\perp$ sua projeção ortogonal na superfície de decisão, tal que:

$$\mathbf{x} = \mathbf{x}_\perp + r\frac{\mathbf{w}}{\|\mathbf{w}\|} \tag{2.26}$$

Multiplicando ambos os lados deste resultado por $\mathbf{w}^{\mathrm{T}}$ e adicionando w_0, utilizando (2.24) e $y(\mathbf{x}_\perp) = \mathbf{w}^{\mathrm{T}}\mathbf{x}_\perp + w_0 = 0$, temos:

$$r = \frac{y(\mathbf{x})}{\|\mathbf{w}\|} \tag{2.27}$$

2.6.2 Função Discriminante para Múltiplas Classes

Bishop (1995) considera a extensão dos discriminantes lineares para $K > 2$ classes. Podemos construir um discriminante de K–classes combinando um número necessário de funções discriminantes de duas classes. Entretanto, isto pode levar a muitas dificuldades (DUDA, HART & STORK, 2000), como será mostrado.

Considere o uso de $K - 1$ classificadores, cada qual resolvendo um problema de duas classes de separação de pontos em uma classe particular C_k, a partir dos pontos que não estão naquela classe. Este é conhecido como classificador um contra o resto (*one-*

versus-the-rest). A Figura 2.3 mostra um exemplo envolvendo três classes, aonde esta abordagem conduz a regiões do espaço de entrada que são classificados de forma ambígua.

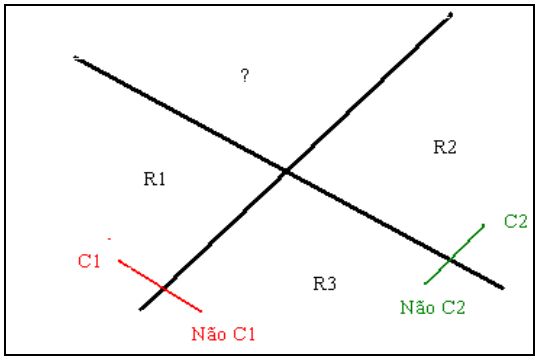

Figura 2.3: Utilização de dois discriminantes para distinguir pontos na classe C_k a partir de pontos que não estão na classe C_k

Uma alternativa é introduzir $K(K - 1)/2$ funções discriminantes binárias, uma para cada par de classe possível. Cada ponto é, então, classificado de acordo com a votação da maioria entre as funções discriminantes. Entretanto, neste caso, também haverá o problema das regiões ambíguas, como ilustrado na Figura 2.4.

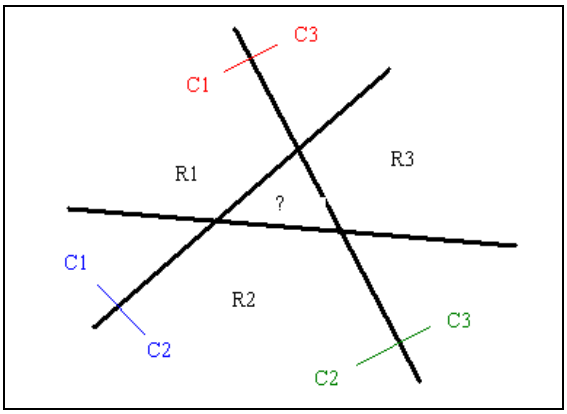

Figura 2.4: Utilização de três discriminantes onde cada qual é utilizado para separar um par de classes C_k e C_j

Podemos evitar estas dificuldades considerando um único discriminante de K classes, compreendendo K funções lineares na forma:

$$y_k(\mathbf{x}) = \mathbf{w}_k^{\mathrm{T}}\mathbf{x} + w_{k0} \qquad \textbf{(2.28)}$$

e então associar um ponto $\mathbf{x}$ a uma classe C_k, se $y_k(\mathbf{x}) > y_j(\mathbf{x})$, para todo $j \neq k$. O limite de decisão entre as classes C_k e C_j é, então, determinado por $y_k(\mathbf{x}) = y_j(\mathbf{x})$ e consequentemente corresponde a um hiperplano $(D - 1)$-dimensional definido por:

$$\left(\mathbf{w}_{\mathrm{k}} - \mathbf{w}_{\mathrm{j}}\right)^{\mathrm{T}} \mathbf{x} + \left(w_{\mathrm{k0}} - w_{\mathrm{j0}}\right) = 0 \qquad \textbf{(2.29)}$$

Este tem a mesma forma do limite de decisão para duas classes, como exposto na seção 2.6.1, e então, propriedades geométricas análogas.

As regiões de decisão de cada discriminante são sempre simplesmente conectadas e convexas. Para representar isto, considere dois pontos $\mathbf{x}_A$ e $\mathbf{x}_B$, ambos arbitrários na região de decisão R_k, como ilustrado na Figura 2.5.

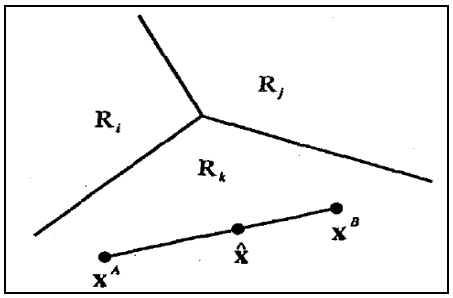

Figura 2.5: Regiões de Decisão para um Discriminante Linear Multiclasse com limites de decisão apresentados em vermelho

Qualquer ponto $\hat{\mathrm{x}}$ arbitrário sobre a linha que conecta $\mathbf{x}_A$ e $\mathbf{x}_B$ pode ser expresso na forma:

$$\hat{\mathbf{x}} = \lambda \mathbf{x}_{\mathrm{A}} + (1 - \lambda)\mathbf{x}_{\mathrm{B}} \qquad \textbf{(2.30)}$$

Onde $0 \leq \lambda \leq 1$. A partir da linearidade das funções discriminantes, segue que:

$$y_k(\hat{\mathbf{x}}) = \lambda y_k(\mathbf{x}_{\mathrm{A}}) + (1 - \lambda) y_k(\mathbf{x}_{\mathrm{B}}) \qquad \textbf{(2.31)}$$

Devido ambos, $\mathbf{x}_A$ e $\mathbf{x}_B$, estarem arbitrados dentro de R_k, segue que $y_k(\mathbf{x_A}) > y_j(\mathbf{x_A})$ e $y_k(\mathbf{x_B}) > y_j(\mathbf{x_B})$, para todo $j \neq k$, e consequentemente $y_k(\hat{\mathrm{x}}) > y_j(\hat{\mathrm{x}})$, sendo $\hat{\mathrm{x}}$ também arbitrado dentro da região R_k. Assim R_k é simplesmente conectada e convexa.

Deve-se notar que para duas classes, podemos empregar o formalismo discutido aqui, baseado em duas funções discriminantes $y_1(\mathbf{x})$ e $y_2(\mathbf{x})$, ou senão, usar a formulação mais simples, descrita na seção 2.6.1, baseada em uma função discriminante única $y(\mathbf{x})$.

Agora, vamos explorar duas abordagens de aprendizado de parâmetros de funções lineares discriminantes: baseadas em mínimos quadrados e discriminantes lineares de Fisher.

2.6.3 Método dos Mínimos Quadrados para Classificação

Em modelos lineares de regressão sabe-se que minimização da função erro da soma dos quadrados nos conduz a uma solução simples de forma aproximada para os valores de parâmetros. Vamos apresentar a prova que podemos aplicar o mesmo formalismo em problemas de classificação (DE CAMPOS, 2001).

Considere um problema geral de classificação com K classes, com 1 dos K esquemas binários de codificação para o vetor de saídas $\mathbf{t}$. Uma justificativa para usar mínimos quadrados em tal contexto é que ele aproxima a esperança condicional $E[\mathbf{t}|\mathbf{x}]$ do vetor de saída dado o vetor de entrada. Para o esquema de codificação binário, esta esperança condicional é dada pelo vetor de probabilidades da classe posterior. Estas probabilidades contidas no vetor são bastante aproximadas. As aproximações podem ter valores fora do intervalo (0, 1), devido a flexibilidade limitada do modelo linear.

Cada classe C_k é descrita pelo seu próprio modelo linear tal que:

$$y_k(\mathbf{x}) = \mathbf{w}_k^{\mathrm{T}}\mathbf{x} + w_{k0} \tag{2.32}$$

onde $k = 1, 2, ..., K$. Podemos convenientemente agrupá-las usando a notação vetorial tal que:

$$y(\mathbf{x}) = \widetilde{\mathbf{W}}^{\mathrm{T}}\widetilde{\mathbf{x}} \tag{2.33}$$

onde $\widetilde{\mathbf{W}}$ é a matriz cuja k-ésima coluna compreende o vetor $(D + 1)$-dimensional $\widetilde{\mathbf{w}}_k = (w_{k0}, \mathbf{w}_k^{\mathrm{T}})^{\mathrm{T}}$ e $\widetilde{\mathbf{x}} = \begin{pmatrix} 1 \\ \mathbf{x} \end{pmatrix} = \left(1, \mathbf{x}^T\right)^T$, com a entrada adicional x_0=1. Uma nova entrada $\mathbf{x}$ é então associada à classe na qual a saída $y_k = \widetilde{\mathbf{w}}_k^{\mathrm{T}}\widetilde{\mathbf{x}}$ é maior. Determinamos o parâmetro matriz $\widetilde{\mathbf{W}}$ por minimização da função erro da soma dos quadrados.

Para isto considere um conjunto de dados de treinamento $\{\mathbf{x}_n, \mathbf{t}_n\}$, e defina uma matriz $\mathbf{T}$ cuja n-ésima linha é $\widetilde{\mathbf{t}}_n^{\mathrm{T}}$ com uma matriz $\widetilde{\mathbf{W}}$ cuja n-ésima linha é $\widetilde{\mathbf{x}}_n^{\mathrm{T}}$. A função erro da soma dos quadrados pode ser escrita como (para maiores detalhes, analisar Bishop (2006)):

$$E_D(\widetilde{\mathbf{W}}) = \frac{1}{2} Traço\left\{\left(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T}\right)^{\mathrm{T}}\left(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T}\right)\right\} \tag{2.34}$$

Derivando a função em relação a $\widetilde{\mathbf{W}}$, igualada a zero e rearranjando, obtemos a solução para $\widetilde{\mathbf{W}}$ na forma:

$$\widetilde{\mathbf{W}} = \left(\widetilde{\mathbf{X}}^{\mathrm{T}}\widetilde{\mathbf{X}}\right)^{-1}\widetilde{\mathbf{X}}^{\mathrm{T}}\mathbf{T} = \widetilde{\mathbf{X}}^{\lozenge}\mathbf{T} \tag{2.35}$$

onde $\widetilde{\mathbf{X}}^{\Diamond}$ é a pseudo-inversa da matriz $\widetilde{\mathbf{X}}$. Assim, obtemos a função discriminante na forma:

$$y(\mathbf{x}) = \widetilde{\mathbf{W}}^{\mathrm{T}}\widetilde{\mathbf{x}} = \mathbf{T}^{\mathrm{T}}(\widetilde{\mathbf{X}}^{\Diamond})^{\mathrm{T}}\widetilde{\mathbf{x}} \tag{2.36}$$

A abordagem dos mínimos quadrados nos dá uma solução de forma aproximada para os parâmetros da função discriminante. Porém, até mesmo como função discriminante, a abordagem sofre com alguns problemas graves. Na solução por mínimos quadrados falta robustez nas camadas de saída e isto se aplica igualmente na aplicação de classificação (HAYKIN, 1999) (DE CAMPOS, 2001).

2.6.4 Análise Discriminante Linear de Fisher (*LDA*)

Segundo Bishop (2006), uma maneira de ver um modelo de classificação linear é em termos de redução de dimensionalidade. Considere primeiro um caso de duas classes, e suponha tomar um vetor de entrada **x** *D*-dimensional e projetá-lo para uma dimensão usando:

$$y = \mathbf{w}^{\mathrm{T}}\mathbf{x} \tag{2.37}$$

Se adicionarmos um *threshold* em y e classificarmos $y \geq -w_0$ como classe C_1, caso contrário classe C_2, então obtemos o classificador padrão discutido na seção anterior. Em geral, a projeção em uma dimensão conduz a uma perda considerável de informação, e classes que estão bem separadas no espaço original *D*-dimensional podem tornar-se fortemente sobrepostos em uma dimensão. Porém, ao ajustar os componentes do vetor de pesos **w**, podemos selecionar uma projeção que maximize a separação das classes.

Bishop (1995) considera um problema de duas classes na qual existem N_1 pontos da classe C_1 e N_2 pontos da classe C_2, tal que os vetores médios das duas classes são dados por:

$$\mathbf{m}_1 = \frac{1}{N_1}\sum_{n\in C_1}\mathbf{x}_n, \qquad \mathbf{m}_2 = \frac{1}{N_2}\sum_{n\in C_2}\mathbf{x}_n, \tag{2.38}$$

A medida mais simples de separação das classes, quando projetadas sobre **w**, é a separação das classes médias projetadas. Isto sugere que podemos escolher **w** para maximizar:

$$m_2 - m_1 = \mathbf{w}^{\mathrm{T}}\left(\mathbf{m}_2 - \mathbf{m}_1\right) \tag{2.39}$$

onde

$$m_{\mathrm{k}} = \mathbf{w}^{\mathrm{T}}\mathbf{m}_{\mathrm{k}} \tag{2.40}$$

é a média dos dados projetados a partir da classe C_k.

Porém, ainda existe um problema com esta abordagem, como mostrado na Figura 2.6. Esta figura mostra duas classes que estão bem separadas em um espaço bidimensional original (x_1, x_2). Vemos que a projeção sobre o eixo x_1 apresenta uma separação muito maior das médias das classes projetadas em relação à projeção sobre o eixo x_2. Entretanto, a separação dos dados projetados é melhor quando os dados são projetados sobre o eixo x_2. Esta dificuldade surge devido às covariâncias fortemente não-diagonais das distribuições das classes.

A ideia proposta por Fisher é maximizar a função que realizará a maior separação entre as classes médias projetadas enquanto proporciona uma variância pequena dentro de cada classe, minimizando assim, a sobreposição de classes.

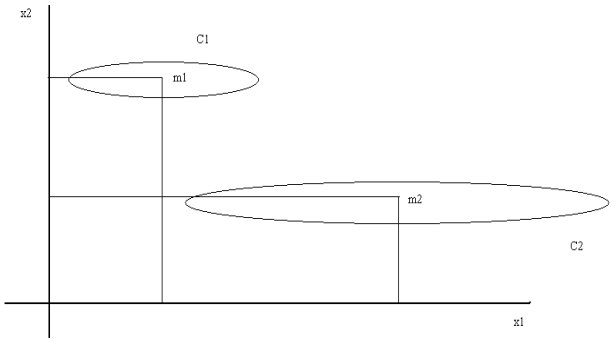

Figura 2.6: Projeção de duas amostras sobre os eixos x_1 e x_2.

A fórmula de projeção (2.37) transforma o conjunto de pontos de dados classificados em **x** para um conjunto classificado em um espaço unidimensional y. A variância intraclasse dos dados transformados a partir da classe C_k é então, dada por:

$$s_k^2 = \sum_{n \in C_k} (y_n - m_k)^2 \qquad \textbf{(2.41)}$$

onde $y_n = \mathbf{w}^{\mathrm{T}} \mathbf{x}_n$. Podemos definir que a variância total intraclasse para todo o conjunto de dados é simplesmente $s_1^2 + s_2^2$. O critério de Fisher é definido para ser a razão da variância entre classes pela variância intraclasse e é dado por:

$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} \qquad \textbf{(2.42)}$$

Podemos realizar explicitamente a dependência sobre **w** usando (2.37), (2.40) e (2.41) para reescrever o critério de Fisher na forma:

$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{T}} \mathbf{S}_{\mathrm{B}} \mathbf{w}}{\mathbf{w}^{\mathrm{T}} \mathbf{S}_{\mathrm{W}} \mathbf{w}} \qquad \textbf{(2.43)}$$

onde $\mathbf{S}_B$ é a matriz de covariância entre classes e é dada por

$$\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T \tag{2.44}$$

e $\mathbf{S}_W$ é a matriz de covariância intraclasse, dada por:

$$\mathbf{S}_W = \sum_{n \in C_1} (\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^T + \sum_{n \in C_2} (\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^T \tag{2.45}$$

Derivando (2.43) em relação a $\mathbf{w}$, encontramos que $J(\mathbf{w})$ é máximo quando:

$$(\mathbf{w}^T \mathbf{S}_B \mathbf{w}) \mathbf{S}_W \mathbf{w} = (\mathbf{w}^T \mathbf{S}_W \mathbf{w}) \mathbf{S}_B \mathbf{w} \tag{2.46}$$

A partir de (2.44), podemos observar que $\mathbf{S}_B\mathbf{w}$ é sempre na direção de $(\mathbf{m}_2 - \mathbf{m}_1)$. Além disso, não nos preocupamos com a magnitude de $\mathbf{w}$, somente com sua direção, e então desconsideramos os fatores escalares $(\mathbf{w}^T\mathbf{S}_B\mathbf{w})$ e $(\mathbf{w}^T\mathbf{S}_W\mathbf{w})$. Multiplicando ambos os lados de (2.46) por S_W^{-1}, então obtemos:

$$\mathbf{w} \propto \mathbf{S}_W^{-1}(\mathbf{m}_2 - \mathbf{m}_1) \tag{2.47}$$

O resultado (2.47) é conhecido como *Discriminante Linear de Fisher*, embora estritamente isto não seja um discriminante, mas uma escolha específica de direção para projeção dos dados para até uma dimensão. Entretanto, os dados projetados podem subsequentemente ser utilizados para construir um discriminante, escolhendo y_0 tal que classifique um novo ponto como pertencente a C_1 se $y(\mathbf{x}) \geq y_0$ e classifique como pertencente a C_2, caso contrário.

2.6.5 Análise Multiclasse Discriminante Linear de Fisher

Bishop (2006) considera a generalização do discriminante de Fisher para $K > 2$ classes, e assume que a dimensionalidade D do espaço de entrada é maior que o número K de classes. A seguir, introduz $D' > 1$ "características" lineares $y_k = \mathbf{w}_k^T\mathbf{x}$, onde $k = 1$, ..., D'. Estes valores característicos podem convenientemente ser agrupados para formar o vetor $\mathbf{y}$. Similarmente, o vetor de pesos $\{\mathbf{w}_k\}$ pode ser considerado as colunas da matriz $\mathbf{W}$, tal que:

$$\mathrm{y} = \mathbf{W}^T\mathbf{x} \tag{2.48}$$

A generalização da matriz de covariância intraclasse para o caso de K classes a partir de (2.45) é dada por:

$$\mathbf{S}_W = \sum_{k=1}^{K} \mathbf{S}_k \tag{2.49}$$

onde

$$\mathbf{S}_k = \sum_{n \in C_k} (\mathbf{x}_n - \mathbf{m}_k)(\mathbf{x}_n - \mathbf{m}_k)^{\mathrm{T}} \tag{2.50}$$

$$\mathbf{m}_k = \frac{1}{N_k} \sum_{n \in C_k} \mathbf{x}_n \tag{2.51}$$

e N_k é o número de padrões na classe C_k. Para encontrar uma generalização da matriz de covariância entre classes, seguimos (DUDA, HART & STORK, 2000) e consideramos primeiro a matriz de covariância total

$$\mathbf{S}_{\mathrm{T}} = \sum_{n=1}^{N} (\mathbf{x}_n - \mathbf{m})(\mathbf{x}_n - \mathbf{m})^{\mathrm{T}} \tag{2.52}$$

onde **m** é o ponto médio do conjunto total de dados

$$\mathbf{m} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n = \frac{1}{N} \sum_{n=1}^{N} \mathrm{N}_k \mathbf{m}_k \tag{2.53}$$

e $\mathrm{N} = \sum_k \mathrm{N}_k$ é o número total de pontos de dados. A matriz de covariância total pode ser decomposta na soma da matriz de covariância intraclasse, dada por (2.49) e (2.50), mais uma matriz adicional $\mathbf{S}_{\mathrm{B}}$, na qual identificamos como uma matriz de covariância entre classes

$$\mathbf{S}_{\mathrm{T}} = \mathbf{S}_{\mathrm{W}} + \mathbf{S}_{\mathrm{B}} \tag{2.54}$$

Onde

$$\mathbf{S}_{\mathrm{B}} = \sum_{k=1}^{K} N_k (\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^{\mathrm{T}} \tag{2.55}$$

Estas matrizes de covariância foram definidas no espaço original **x**. Podemos agora definir matrizes similares em um espaço projetado D'-dimensional

$$\mathbf{s}_{\mathrm{W}} = \sum_{k=1}^{K} \sum_{n \in C_k} (\mathbf{y}_n - \boldsymbol{\mu}_k)(\mathbf{y}_n - \boldsymbol{\mu}_k)^{\mathrm{T}} \tag{2.56}$$

e

$$\mathbf{s}_{\mathrm{B}} = \sum_{k=1}^{K} N_k (\boldsymbol{\mu}_k - \boldsymbol{\mu})(\boldsymbol{\mu}_k - \boldsymbol{\mu})^{\mathrm{T}} \tag{2.57}$$

onde

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n \in C_k} \mathbf{y}_n, \qquad \boldsymbol{\mu} = \frac{1}{N} \sum_{k=1}^{K} \mathrm{N}_k \boldsymbol{\mu}_k \tag{2.58}$$

Agora existem muitas possibilidades de escolha de critérios (FUKUNAGA, 1990). Um exemplo é dado por:

$$J(\mathbf{W}) = Traço\left\{\mathbf{s}_{\mathrm{W}}^{-1} \mathbf{s}_{\mathrm{B}}\right\} \tag{2.59}$$

Este critério pode ser reescrito como uma função explícita da matriz projeção **W** na forma:

$$J(\mathbf{w}) = Traço\left\{\left(\mathbf{W}\mathbf{S}_{\mathrm{W}}\mathbf{W}^{\mathrm{T}}\right)^{-1}\left(\mathbf{W}\mathbf{S}_{\mathrm{B}}\mathbf{W}^{\mathrm{T}}\right)\right\} \qquad \textbf{(2.60)}$$

Capítulo III – Redes Neurais Artificiais

Neste capítulo serão apresentados os seguintes itens: o paradigma dos modelos neurais, os algoritmos de aprendizado, os conceitos fundamentais de redes neurais artificiais e detalhes de um modelo específico de abordagem: a Rede Neural *Perceptron* Multicamadas (*MLP – MultiLayer Perceptron*) com treinamento realizado pelo algoritmo *Backpropagation*, de interesse específico para esta pesquisa.

3.1 Introdução às Redes Neurais

Redes Neurais Artificiais (*RNAs*) são sistemas paralelos e distribuídos compostos por unidades de processamento simples (nodos ou neurônios ou ainda, unidades) que calculam determinadas funções matemáticas (normalmente não-lineares). Tais unidades são dispostas em uma ou mais camadas e interligadas por um grande número de conexões, geralmente unidirecionais. Na maioria dos modelos estas conexões estão associadas a pesos, os quais armazenam o conhecimento representado no modelo e servem para ponderar a entrada recebida por cada neurônio da rede.

A solução de problemas através de *RNAs* é bastante atrativa, já que a forma como estes são representados internamente pela rede e o paralelismo natural inerente à arquitetura das *RNAs* criam a possibilidade de um desempenho superior ao dos modelos convencionais. Em *RNAs*, o procedimento usual na solução de problemas passa inicialmente por uma fase de aprendizagem, em que um conjunto de exemplos é apresentado para a rede, a qual extrai automaticamente as características necessárias para representar a informação fornecida. Estas características são utilizadas posteriormente para gerar respostas para o problema.

A capacidade de aprender através de exemplos e de generalizar a informação aprendida é o atrativo principal da solução de problemas através de *RNAs*. A generalização, que está associada à capacidade da rede aprender através de um conjunto reduzido de exemplos e posteriormente dar respostas coerentes para dados não-conhecidos, é uma demonstração de que a capacidade das *RNAs* vai muito além do que

simplesmente mapear relações de entrada e saída. As *RNAs* são capazes de extrair informações não-apresentadas de forma explícita através dos exemplos.

As *RNAs* tentam reproduzir as funções das redes biológicas, buscando implementar seu comportamento básico e sua dinâmica. No entanto, do ponto de vista físico, no momento as redes artificiais se diferenciam bastante das redes biológicas. É importante, contudo observar as similaridades entre estes dois tipos de sistemas, tanto para que se possa entender melhor o sistema nervoso quanto para buscar ideias e inspirações para a pesquisa em neurocomputação. Como características comuns, pode-se citar que os dois sistemas são baseados em unidades de computação paralela e distribuída que se comunicam por meio de conexões sinápticas, possuem detectores de características, redundância e modularização das conexões.

3.2 Neurônios Artificiais – Modelo *MCP*

O modelo *MCP* de neurônio proposto por McCulloch & Pitts (MCCULLOCH & PITTS, 1943) é uma simplificação do que se sabia até então a respeito do neurônio biológico. Sua descrição matemática resultou em um modelo com n terminais de entrada x_1, x_2, ..., x_n (que representam os dendritos) e apenas um terminal de saída y (representando o axônio).

Para simular o comportamento das sinapses, os terminais de entrada do neurônio têm pesos acoplados w_1, w_2, ..., w_n cujos valores podem ser positivos ou negativos, dependendo das sinapses correspondentes serem inibitórias ou excitatórias. O efeito de uma sinapse particular i no neurônio pós-sináptico é dado por $x_i w_i$. Os pesos determinam em que grau o neurônio deve considerar sinais de disparo que ocorrem naquela conexão. Uma descrição do modelo está ilustrada na Figura 3.1.

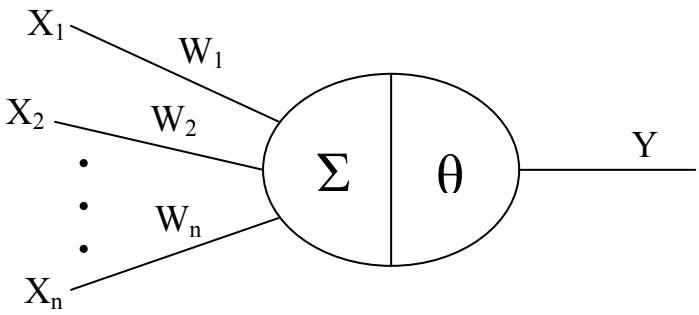

Figura 3.1: Neurônio de McCulloch e Pitts

Um neurônio biológico dispara quando a soma dos impulsos que ele recebe ultrapassa o seu limiar de excitação (*threshold*). O corpo do neurônio, por sua vez, é emulado por um mecanismo simples que faz a soma dos valores $x_i w_i$ recebidos pelo neurônio (soma ponderada) e decide se o neurônio deve ou não disparar (saída igual a 1 ou a 0) comparando a soma obtida ao limiar do neurônio. No modelo *MCP*, a ativação do neurônio é obtida através da aplicação de uma "função de ativação", que ativa ou não a saída, dependendo do valor da soma ponderada das suas entradas. Na descrição original do modelo *MCP*, a função de ativação é dada pela função de limiar e o nodo *MCP* terá então a sua saída ativa quando:

$$\sum_{i=1}^{n} x_i w_i \geq \theta \qquad \textbf{(3.1)}$$

Onde n é o número de entradas do neurônio, w_i é o peso associado à entrada x_i e θ é o limiar do neurônio.

A partir do modelo proposto por McCulloch & Pitts, foram definidos vários outros modelos que permitem a produção de uma saída qualquer, não necessariamente zero ou um, e com diferentes funções de ativação, por exemplo: função linear, função rampa, função degrau e função sigmoidal.

3.3 Algoritmos de Aprendizado

Redes neurais artificiais possuem a capacidade de aprender por exemplos e fazer interpolações e extrapolações do que aprenderam. Um conjunto de procedimentos definidos para adaptar os parâmetros de uma *RNA* para que a mesma possa *aprender* uma determinada função é chamado de *algoritmo de aprendizado*. Existe uma diversidade de algoritmos de aprendizado e estes diferem basicamente pela maneira na qual o ajuste de pesos é realizado (BRAGA, CARVALHO & LUDERMIR, 2000).

A etapa de aprendizagem consiste em um processo interativo de ajuste de parâmetros da rede, os pesos das conexões entre as unidades de processamento, que guardam, ao final do processo, o conhecimento que a rede adquiriu do ambiente em que está operando. Diversos métodos para treinamento de rede foram desenvolvidos, podendo ser agrupados em dois paradigmas principais: s*upervisionado* e *não-supervisionado*.

3.3.1 Aprendizado Supervisionado

É assim chamado porque a entrada e a saída desejadas para a rede são fornecidas por um supervisor externo. O objetivo é ajustar os parâmetros da rede, de forma a encontrar uma ligação entre os pares de entrada e saída fornecidos. A Figura 3.2 ilustra o mecanismo de aprendizado supervisionado.

O supervisor indica explicitamente um comportamento bom ou ruim para a rede, visando direcionar o processo de treinamento. A rede tem a sua saída calculada comparada com a saída desejada, recebendo informações do supervisor sobre o erro da resposta atual, objetivando ajustar os pesos das conexões para minimizar o erro. A minimização da diferença é incremental, uma vez que pequenos ajustes são feitos nos pesos a cada etapa de treinamento, de tal forma que estes caminhem, se houver solução possível, para uma solução. A soma dos erros quadráticos de todas as saídas é normalmente utilizada como medida de desempenho da rede e também como função de custo a ser minimizada pelo algoritmo de treinamento.

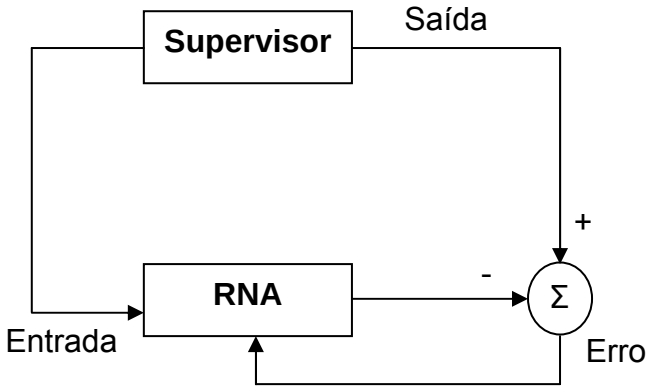

Figura 3.2: Aprendizado Supervisionado

Os exemplos mais conhecidos de algoritmos para de aprendizado supervisionado são a regra delta (WIDROW& HOFF, 1960) e a sua generalização para redes de múltiplas camadas, o algoritmo *backpropagation* (RUMELHART, HINTON *et al.*, 1986) que serão detalhados posteriormente neste trabalho.

A adaptação por correção de erros procura minimizar a diferença entre a soma ponderada das entradas pelos pesos e a saída desejada, ou seja, o erro da resposta atual da rede. O termo $e(t)$ do erro deve ser escrito como:

$$e(t) = d(t) - y(t) \qquad \textbf{(3.2)}$$

Onde $d(t)$ é a saída desejada e $y(t)$ é a resposta atual calculada no instante de tempo t. A forma genérica para alteração dos pesos por correção de erros é apresentada a seguir:

$$w_i(t+1) = w_i(t) + \eta e(t) x_i(t) \quad \textbf{(3.3)}$$

Onde η é a taxa de aprendizado e $x_i(t)$ é a entrada do neurônio i para o tempo t. Isto é, o ajuste dos pesos deve ser proporcional ao produto do erro pelo valor da entrada da sinapse naquele instante de tempo.

3.3.2 Aprendizado Não-Supervisionado

No aprendizado não-supervisionado não há um supervisor para acompanhar o processo de aprendizado. Este método está ilustrado na Figura 3.3.

Para estes algoritmos, somente os padrões de entrada estão disponíveis para a rede, ao contrário do aprendizado supervisionado, cujo conjunto de treinamento possui pares de entrada e saída. A partir do momento em que a rede estabelece uma harmonia com as regularidades estatísticas da entrada de dados, desenvolve-se nela uma habilidade de formar representações internas para codificar características da entrada e criar novas classes ou grupos automaticamente. É importante salientar que este tipo de aprendizado só se torna possível quando existe redundância nos dados de entrada. Sem redundância seria impossível encontrar quaisquer padrões ou características dos dados de entrada.

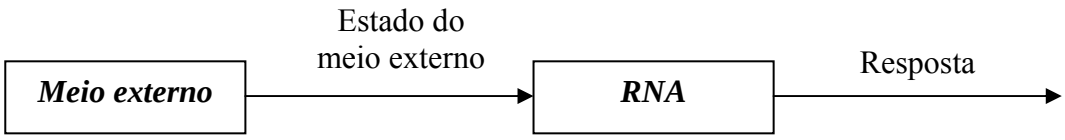

Figura 3.3: Aprendizado Não-supervisionado

Alguns métodos de implementação de aprendizado não-supervisionado são: Aprendizado Hebbiano, Modelo de Linsker, Regra de Oja, Regra de Yuille e Aprendizado por Competição. Pelo motivo deste trabalho fazer uso do aprendizado supervisionado, estes modelos não serão detalhados.

3.4 Redes *Perceptron* Multicamadas com Algoritmo *Backpropagation*

Quando Redes Neurais Artificiais de uma só camada são utilizadas os padrões de treinamento apresentados à entrada são mapeados diretamente em um conjunto de

padrões de saída da rede, ou seja, não é possível a formação de uma representação interna. Neste caso, a codificação proveniente do mundo exterior deve ser suficiente para implementar esse mapeamento.

Tal restrição implica que padrões de entrada similares resultem em padrões de saída similares, o que leva o sistema à incapacidade de aprender importantes mapeamentos. Como resultado, padrões de entrada com estruturas similares, fornecidos do mundo externo, que levem a saídas diferentes não são possíveis de serem mapeados por redes sem representações internas, isto é, sem camadas intermediárias.

Minsky & Papert (MINSKY & PAPERT, 1969) analisaram matematicamente o *Perceptron* e demonstraram que redes de uma camada não são capazes de solucionar problemas que não sejam linearmente separáveis. Como eles não acreditavam na possibilidade de se construir um método de treinamento para redes com mais de uma camada, eles concluíram que as redes neurais seriam sempre suscetíveis a essa limitação.

Contudo, o desenvolvimento do algoritmo de treinamento *backpropagation*, por Rumelhart, Hinton & Williams, em 1986 (RUMELHART, HINTON *et al.*, 1986), precedido por propostas semelhantes ocorridas nos anos 70 e 80, mostrou que é possível treinar eficientemente redes com camadas intermediárias, resultando no modelo de Redes Neurais Artificiais mais utilizado atualmente, as redes *Perceptron* Multicamadas (*MultiLayer Perceptron* - *MLP*), treinadas com o algoritmo *backpropagation*.

Nessas redes, cada camada tem uma função específica. A camada de saída recebe os estímulos da camada intermediária e constrói o padrão que será a resposta. As camadas intermediárias funcionam como extratoras de características, seus pesos são uma codificação de características apresentadas nos padrões de entrada e permitem que a rede crie sua própria representação, mais rica e complexa, do problema.

3.4.1 Portas *Threshold* e Redes *Perceptron*

As portas *threshold* (do tipo limiar) podem ser divididas em três tipos: linear, quadrática e polinomial (MUROGA, 1971). A função executada por cada uma delas é basicamente a mesma: comparação da soma ponderada das entradas com um valor de *threshold*. Caso a soma exceda o *threshold*, a saída é ativada, permanecendo desativada em caso contrário. No entanto, estes modelos diferem entre si pela complexidade com que seus pesos são calculados. Quanto mais complexos os termos associados a cada um dos pesos, mais complexas as superfícies que podem ser formadas no espaço η-

dimensional e maior flexibilidade possui a porta *threshold* na solução do problema de mapeamento.

As **portas *threshold* lineares** são definidas de forma semelhante ao nodo *MCP*, conforme mostrado na Figura 3.4:

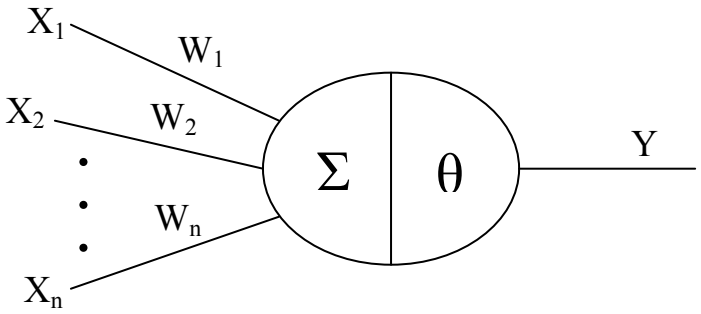

Figura 3.4: Porta *Threshold* Linear

$$y = \begin{cases} 1, \sum w_i x_i \geq \theta \\ 0, \sum w_i x_i < \theta \end{cases} \quad \textbf{(3.4)}$$

As portas *threshold* lineares estão restritas à solução de problemas que sejam linearmente separáveis, ou seja, a problemas cuja solução pode ser obtida pela separação de duas regiões por meio de uma reta ou um hiperplano para o caso η-dimensional.

Para valores grandes de η, a relação entre o número de funções linearmente separáveis e o número total de funções booleanas tende a zero, restringindo assim a utilização de portas lineares (HASSOUN, 1995). Para aumentar a capacidade computacional das mesmas, as **portas *threshold* quadráticas** são definidas conforme Figura 3.5:

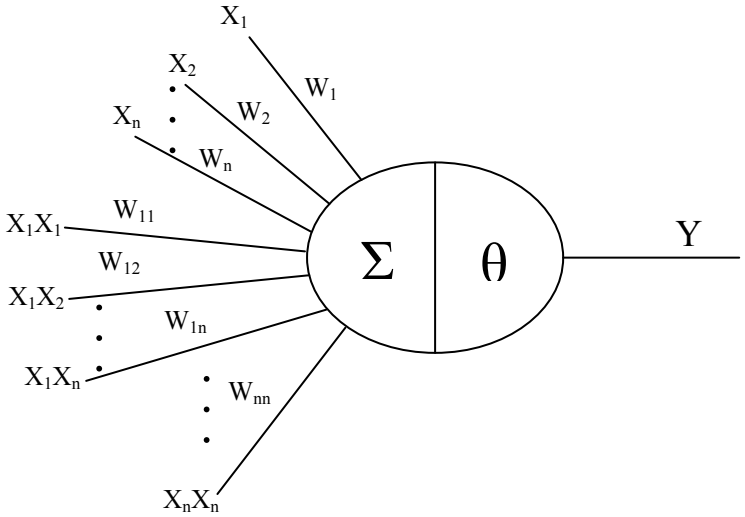

Figura 3.5: Porta *Threshold* Quadrática

$$y = \begin{cases} 1, \sum w_i x_i + \sum w_{ij} x_i x_j \geq \theta \\ 0, \sum w_i x_i + \sum w_{ij} x_i x_j < \theta \end{cases} \quad \textbf{(3.5)}$$

O termo adicional na Equação 3.5 confere à porta *threshold* quadrática um maior poder computacional, já que a mesma possui um número maior de parâmetros livres ajustáveis. Como pode ser visto na Equação 3.5, a porta quadrática possui termos quadráticos em cada uma das variáveis de entrada, além de termos com produtos cruzados entre cada uma delas. Os pesos de entrada (parâmetros livres) definem a importância de cada um destes termos na definição da superfície de decisão.

O modelo proposto por Rosenblatt, conhecido como ***Perceptron*** (ROSENBLATT, 1958), é composto por uma estrutura de rede, tendo como unidades básicas nodos *MCP*, e por uma regra de aprendizado. Rosenblatt demonstrou o teorema de convergência do *perceptron*, que mostra que um nodo *MCP* treinado com o algoritmo de aprendizado do *perceptron* sempre converge caso o problema em questão seja linearmente separável.

A topologia original descrita por Rosenblatt é composta por unidades de entrada (retina), por um nível intermediário formado pelas unidades de associação e por um nível de saída formado pelas unidades de resposta. A retina consiste basicamente em unidades sensoras, e as unidades intermediárias de associação, embora formadas por nodos *MCP*, possuem pesos fixos, definidos antes do período de treinamento. A Figura 3.6 mostra um esboço da topologia do *perceptron*.

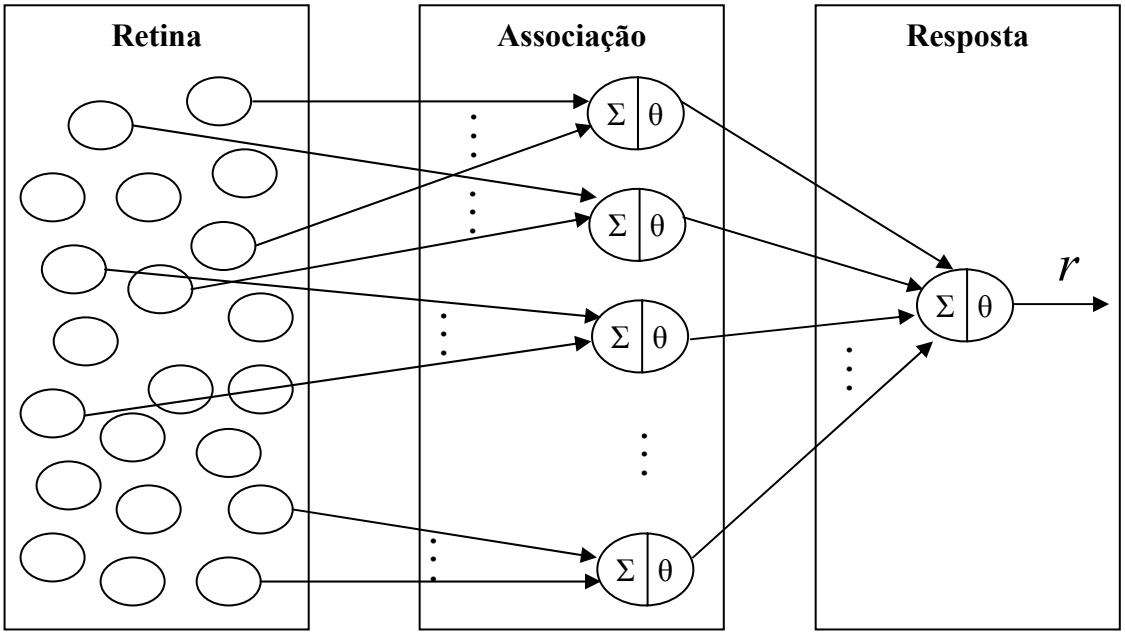

Figura 3.6: Topologia de um *perceptron* simples com uma única saída

3.4.2 Arquitetura de uma Rede *MLP*

Redes *Perceptron* Multicamadas (*MLP – MultiLayer Perceptron*) apresentam um poder computacional muito maior do que aquele apresentado pelas redes sem camadas intermediárias. Ao contrário destas redes, *MLPs* podem tratar com dados que não são linearmente separáveis. Teoricamente, redes com duas camadas intermediárias podem implementar qualquer função, seja ela linearmente separável ou não (CYBENCO, 1989). A precisão obtida e a implementação da função objetivo dependem do número de nodos utilizados nas camadas intermediárias. A Figura 3.7 apresenta uma rede *MLP* típica.

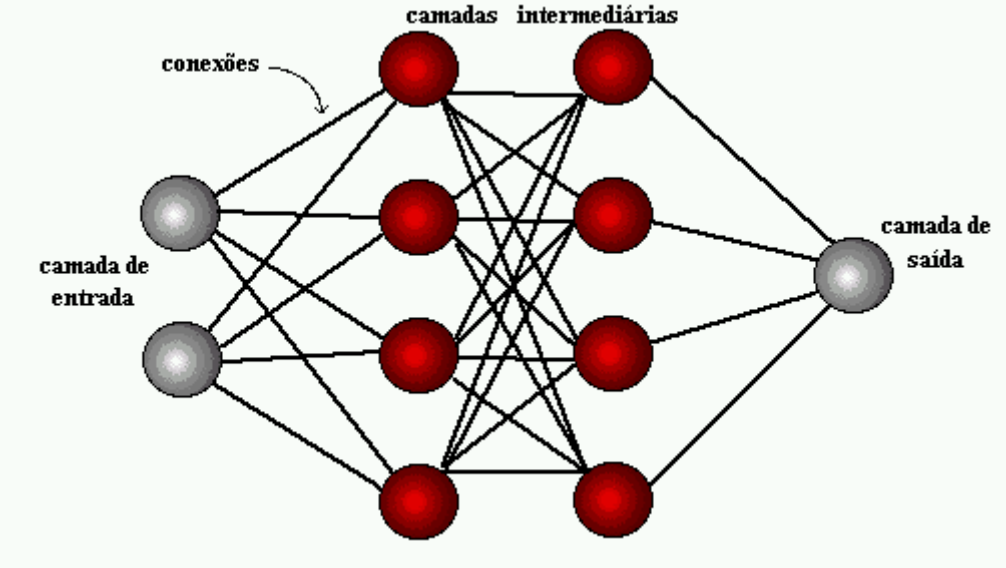

Figura 3.7: Rede *MLP* típica com duas camadas intermediárias

Um dos principais aspectos relacionados ao projeto de redes *MLP* diz respeito à função de ativação utilizada. Diversas funções de ativação têm sido propostas para redes multicamadas. Estas funções são não lineares e diferenciáveis. A função de ativação mais utilizada é a *sigmoide*.

Em uma rede multicamadas, o processamento realizado por cada nodo é definido pela combinação dos processamentos realizados pelos nodos da camada anterior que estão conectados a ele. Quando se segue da primeira camada intermediária em direção à camada de saída, as funções implementadas se tornam cada vez mais complexas. Estas funções definem como é realizada a divisão do espaço de decisão. Para uma rede com pelo menos duas camadas intermediárias, pode-se dizer que o seguinte processamento ocorre em cada uma das camadas:

a) Primeira camada intermediária: cada nodo traça retas no espaço de padrões de treinamento;

b) Segunda camada intermediária: cada nodo combina as retas traçadas pelos neurônios da camada anterior conectados a ele, formando regiões convexas, onde o número de lados é definido pelo número de unidades a ele conectadas;

c) Camada de saída: cada nodo forma regiões que são combinações das regiões convexas definidas pelos nodos a eles conectados da camada anterior

Pode ser dito que as unidades intermediárias de uma rede *MLP* funcionam como detectores de características. Eles geram uma codificação interna dos padrões de entrada, que é então utilizada para a definição da saída da rede. Dado um número suficientemente grande de unidades intermediárias, é possível formar representações internas para qualquer conjunto de padrões de entrada.

Cybenko (CYBENKO, 1989) investigou o número de camadas intermediárias necessárias para a implementação de classes de funções em uma *RNA*. Os resultados indicam que:

- Uma camada intermediária é suficiente para aproximar qualquer função contínua;
- Duas camadas intermediárias são suficientes para aproximar qualquer função matemática.

Deve ser observado, contudo, que em alguns casos a utilização de duas ou mais camadas intermediárias pode facilitar o treinamento da rede. A utilização de um grande número de camadas intermediárias não é recomendada, no entanto, pois, cada vez que o erro medido durante o treinamento é propagado para a camada anterior, ele se torna menos útil e preciso. A única camada que tem uma noção precisa do erro cometido pela rede é a camada de saída.

Com relação ao número de nodos nas camadas intermediárias, este é, em geral, definido empiricamente. Este número depende fortemente da distribuição dos padrões de treinamento e validação da rede. Alguns métodos, no entanto, têm sido propostos. Os mais utilizados são:

- Definir o número de unidades em função do número de entradas e saídas. Deve ser observado que este método não pode ser utilizado de forma genérica (BRAGA, CARVALHO & LUDERMIR, 2000);
- Utilizar um número de conexões dez vezes menor que o número de exemplos. Este método apenas reduz a incidência de *overfitting*. Se o número de exemplos

for muito maior que o número de conexões, *overfitting* é improvável, mas pode ocorrer *underfitting* (a rede não converge durante o seu treinamento) (BRAGA, CARVALHO & LUDERMIR, 2000).

O número adequado de nodos na camada intermediária depende de vários fatores, como: número de exemplos de treinamento; quantidade de ruído presente nos exemplos; complexidade da função a ser aprendida; distribuição estatística dos dados de treinamento. A solução neural mais eficiente é aquela em que o número de unidades cresce polinomialmente (e não exponencialmente) com o aumento do número de unidades de entrada (HAYKIN, 1999) (BISHOP, 1995) (BISHOP, 2006).

Para a solução de problemas práticos de reconhecimento de padrões, aloca-se para a rede um número de unidades intermediárias suficiente para a solução do problema. Deve-se ter cuidado para não utilizar nem unidades demais, o que pode levar a rede a memorizar os padrões de treinamento, em vez de extrair as características gerais que permitirão a generalização ou o reconhecimento de padrões não vistos durante o treinamento (*overfitting*), nem um número muito pequeno, que pode forçar a rede a gastar tempo em excesso tentando encontrar uma representação ótima, isto é, as unidades utilizadas podem ficar sobrecarregadas, tendo que lidar com um elevado número de restrições.

Como na maioria dos casos o conjunto de treinamento de uma *RNA* é composto de dados experimentais, estes contêm implicitamente erros inerentes aos processos de amostragem. Desta forma, a aproximação através de *RNAs* deve ser feita visando à obtenção de uma estrutura que seja capaz de modelar os dados sem modelar o ruído contido neles. Este é um problema conhecido na literatura como "*Bias and Variance Dilemma*" (GEMAN, BIENENSTOCK *et al.*, 1992), que envolve a obtenção de um modelo que não seja muito rígido a ponto de não modelar fielmente os dados, mas que também não seja excessivamente flexível a ponto de modelar também o ruído.

O equilíbrio entre a rigidez e a flexibilidade da rede é obtido por meio de seu dimensionamento. Quanto maior a sua estrutura, maior o número de parâmetros livres ajustáveis e, consequentemente, maior a sua flexibilidade. Porém, quando os dados são apresentados à rede, não se tem real conhecimento de sua complexidade, daí a dificuldade do problema e dimensionamento.

Uma forma de se evitar *overfitting* é a adoção de técnicas de poda (*pruning*) (REED, 1993), que por sua vez, envolvem a eliminação de pesos e nodos irrelevantes para a função executada pela rede. Existem basicamente dois tipos de métodos de

pruning: os métodos baseados na avaliação da sensibilidade da saída e os métodos que envolvem modificações na função de custo (REED, 1993).

É comum, em alguns casos, usar-se também um vetor auxiliar de polarizações. A introdução de polarizações resulta no deslocamento da função de transferência, a partir de sua origem, produzindo assim um efeito similar ao do ajuste do limiar do neurônio e maior rapidez na convergência do processo de aprendizado.

3.4.3 Treinamento de uma Rede *MLP* com o Algoritmo *Backpropagation*

Existem atualmente vários algoritmos para treinar redes *MLP* (RUMELHART & MCCLELLAND, 1986) (FAHLMAN, 1988) (RIEDMILLER, 1994) (PEARLMUTTER, 1992) (BATTITI, 1991) (HAGAN & MENHAJ, 1994). Estes algoritmos são geralmente do tipo supervisionado. De acordo com os parâmetros que eles atualizam os algoritmos para treinamento de redes do tipo *MLP*, podem ser classificados em: estáticos e dinâmicos.

Enquanto os algoritmos estáticos não alterem a estrutura da rede, variando apenas os valores de seus pesos, os algoritmos dinâmicos podem tanto reduzir quanto aumentar o tamanho da rede (número de camadas, número de nodos nas camadas intermediárias e número de conexões). Quando o aprendizado estático é utilizado, a mesma regra de aprendizado é empregada para redes do tipo *MLP* com diferentes tamanhos e formatos. É interessante observar que topologias diferentes podem resolver o mesmo problema.

O algoritmo de aprendizado mais conhecido para treinamento das redes *MLP* é o algoritmo *backpropagation* (RUMELHART & MCCLELLAND, 1986). A maioria dos métodos de aprendizado para *RNAs* do tipo *MLP* utiliza variações deste algoritmo. O algoritmo *backpropagation* é um algoritmo supervisionado que utiliza pares (entrada, saída desejada) para, por meio de um mecanismo de correção de erros, ajustar os pesos da rede.

O treinamento ocorre em duas fases, em cada fase percorre e rede em um sentido. Estas duas fases são chamadas fase *forward* e fase *backward*. A fase *forward* é utilizada para definir a saída da rede para um dado padrão de entrada. A fase *backward* utiliza a saída desejada e a saída fornecida pela rede para atualizar os pesos de suas conexões. A Figura 3.8 ilustra estas duas fases.

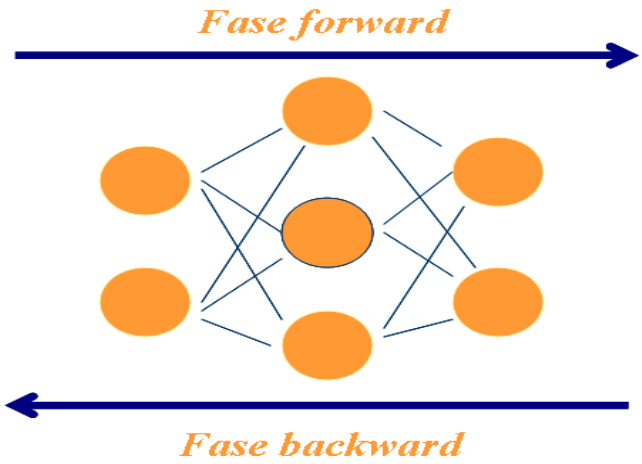

Figura 3.8: Fluxo do processamento do algoritmo *backpropagation*

A fase *forward* é apresentada no algoritmo da Figura 3.9. A fase *backward* é apresentada no algoritmo da Figura 3.10. O algoritmo de *backpropagation*, que faz uso destas duas fases é apresentado na Figura 3.11.

1. A entrada é apresentada à primeira camada da rede, a camada C^0;
2. Para cada camada C^i a partir da camada de entrada
   2.1 Após os nodos da camada C^i ($i > 0$) calcularem seus sinais de saída, estes servem como entrada para a definição das saídas produzidas pelos nodos da camada C^{i+1};
3. As saídas produzidas pelos nodos da última camada são comparadas às saídas desejadas.

Figura 3.9: Sentido *forward* do algoritmo *backpropagation*

1. A partir da última camada, até chegar à camada de entrada;
   1.1 Os nodos da camada atual ajustam seus pesos de forma a reduzir seus erros;
   1.2 O erro de um nodo das camadas intermediárias é calculado utilizando os erros dos nodos da camada seguinte conectados a ele ponderados pelos pesos das conexões entre eles.

Figura 3.10: Sentido *backward* do algoritmo *backpropagation*

```
1. Inicializar os pesos e parâmetros;
2. Repetir até o erro ser mínimo ou até a realização de um dado
número de ciclos:
     2.1 Para cada padrão de treinamento X
          2.1.1 Definir saída da rede através da fase forward;
          2.2.2 Comparar saídas produzidas com as saídas
          desejadas;
          2.2.3 Atualizar os pesos dos nodos através da fase
          backward.
```

Figura 3.11: Descrição do algoritmo *backpropagation*

A Figura 3.12 apresenta a configuração de uma rede *MLP* de 4 camadas (1 de entrada, 2 intermediárias e 1 de saída), treinada com o algoritmo *backpropagation* e utilizada no reconhecimento de 10 comandos, onde a matriz de entrada é de 1000 x 481 (1000 locuções com 481 características cada uma), e a matriz de saída é 1000 x 10. Escolheu-se, para este exemplo, a seguinte topologia: 69 neurônios para a primeira camada intermediária, 20 neurônios para a segunda camada intermediária e 10 neurônios para a camada de saída.

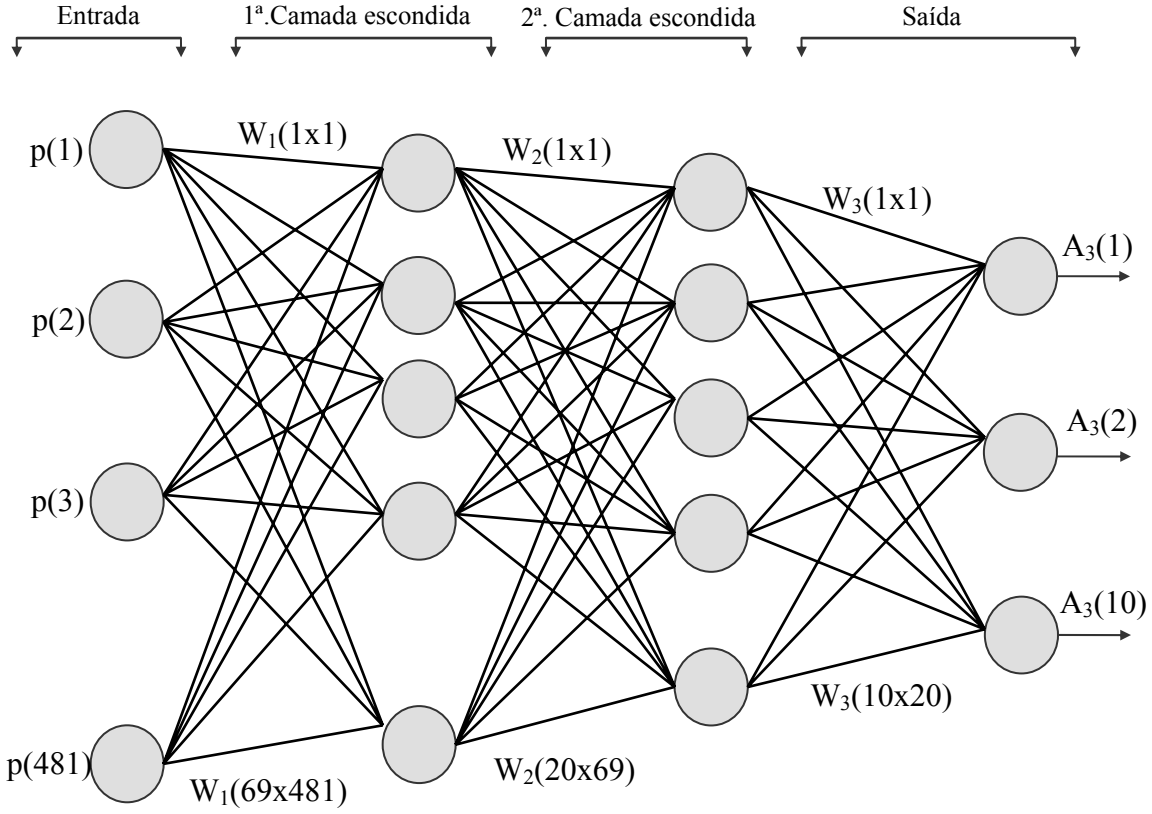

Figura 3.12: Arquitetura da Rede *MLP* com algoritmo *backpropagation*

O algoritmo *backpropagation* é baseado na regra delta proposta por Widrow & Hoff (WIDROW & HOFF, 1960). Este algoritmo propõe uma forma de definir o erro dos nodos das camadas intermediárias, possibilitando o ajuste de seus pesos. Os ajustes os pesos são realizados utilizando-se o método do gradiente.

A definição da regra delta generalizada é semelhante à da regra delta. A função custo a ser minimizada é uma função de erro ou energia, definida pela soma dos erros quadráticos e representada por:

$$E = \frac{1}{2}\sum_{p}\sum_{i=1}^{k}(d_i^p - y_i^p)^2 \qquad \textbf{(3.6)}$$

onde E é a medida de erro total, p é o número de padrões, k é o número de unidades de saída, d_i é a i-ésima saída desejada e y_i é a i-ésima saída gerada pela rede. Esta equação define o erro total cometido pela rede, ou a quantidade em que, para todos os padrões p de um dado conjunto, as saídas geradas pela rede diferem das saídas desejadas.

A regra delta generalizada requer que as funções de ativação utilizadas pelos nodos sejam contínuas, diferenciáveis e, geralmente, não-decrescentes da entrada total recebida pelo nodo. Estas funções são chamadas de funções semilineares. Segue o cálculo do valor de ativação.

$$y_j^p = f_j(net_j^p) \qquad \textbf{(3.7)}$$

Onde:

$$net_j^p = \sum_{i=1}^{n} x_i^p w_{ji} \qquad \textbf{(3.8)}$$

A constante n representa o número de conexões de entrada do nodo j, e w_{ji}, o peso da conexão entre a entrada x_i^p e o nodo j.

Embora o erro total E seja definido pela soma dos erros dos nodos de saída para todos os padrões, será considerado, sem perda de generalidade, que a minimização do erro para cada padrão individualmente levará à minimização do erro total. Assim, o erro passa a ser definido por:

$$E = \frac{1}{2}\sum_{j=1}^{k}(d_j - y_j)^2 \qquad \textbf{(3.9)}$$

A variação dos pesos é definida de acordo com o gradiente descendente do erro com relação ao peso, ou seja, a variação do peso para um dado padrão é definida por:

$$\Delta w_{ji} \propto -\frac{\partial E}{\partial w_{ji}} \qquad \textbf{(3.10)}$$

Agora é necessário definir como cada um dos pesos de cada nodo da rede deve ser ajustado de forma a diminuir o erro total gerado pela rede. Utilizando a regra da cadeia, tem-se que:

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial net_j}\frac{\partial net_j}{\partial w_{ji}} \tag{3.11}$$

Onde $net_j = \sum_{i=1}^{n} x_i w_{ji}$. Então:

$$\frac{\partial net_j}{\partial w_{ji}} = \frac{\partial \sum_{l=1}^{n} x_l w_{jl}}{\partial w_{ji}} = \sum_{l=1}^{n} x_l \delta_{li} = x_i \tag{3.12}$$

A derivada do lado direito da igualdade da Equação 3.11, mede o erro do nodo *j*, e geralmente é abreviada para δ_j .

$$\delta_j = \frac{\partial E}{\partial net_j} \tag{3.13}$$

O cálculo desta derivada também pode ser definido por:

$$\delta_j = \frac{\partial E}{\partial net_j} = \frac{\partial E}{\partial y_l}\frac{\partial y_l}{\partial net_j} \tag{3.14}$$

A derivada $\frac{\partial y_j}{\partial net_j}$, da Equação 3.14 é dada por:

$$\frac{\partial y_j}{\partial net_j} = \frac{\partial f(net_j)}{\partial net_j} = f'(net_j) \tag{3.15}$$

Já o cálculo derivada $\frac{\partial E}{\partial y_j}$ que utilize o erro vai depender da camada onde o nodo *j* se encontra. Se o nodo estiver na última camada, o seu erro pode ser definido utilizando a Equação 3.9. Assim:

$$\frac{\partial E}{\partial y_j} = \frac{\partial (\frac{1}{2}\sum_{i=1}^{k}(d_i - y_i)^2)}{\partial y_j} = \sum_{i=1}^{k}(d_i - y_i)\left(-\frac{\partial y_i}{\partial y_j}\right) = -\sum_{i=1}^{k}(d_i - y_i)\delta_{ij} = -(d_j - y_j) \tag{3.16}$$

Que é a mesma fórmula da regra delta original. Substituindo os dois termos no lado direito da Equação 3.14, obtém-se:

$$\delta_j = -(d_j - y_j) f'(net_j) \tag{3.17}$$

Caso o nodo *j* não seja um nodo de saída, a regra da cadeia é utilizada para escrever:

$$\frac{\partial E}{\partial y_j} = \sum_{l=1}^{M} \frac{\partial E}{\partial net_l} \frac{\partial net_l}{\partial y_j} = \sum_{l=1}^{M} \frac{\partial E}{\partial net_l} \frac{\partial \sum_{i=1}^{n} w_{il} y_i}{\partial y_j} = \sum_{l=1}^{M} \frac{\partial E}{\partial net_l} w_{jl} \quad \textbf{(3.18)}$$

sendo *M* o número total de nodos da camada que se encontra o nodo *j*. Onde:

$$\frac{\partial E}{\partial net_l} = \delta_l \quad \textbf{(3.19)}$$

Substituindo mais uma vez os dois termos do lado direito da Equação 3.14, obtém-se que, para os nodos das camadas intermediárias, o erro é definido por.

$$\delta_j = f'(net_j) \sum_l \delta_l w_{lj} \quad \textbf{(3.20)}$$

Pode-se, assim generalizar a fórmula de ajuste de pesos proposta na Equação 3.10 para:

$$\Delta w_{ji} = \eta \delta_j x_i \quad \textbf{(3.21)}$$

Ou:

$$w_{ji}(t+1) = w_{ji}(t) + \eta \delta_j(t) x_i(t) \quad \textbf{(3.22)}$$

Se o nodo for de saída, o erro δ_j será definido pela Equação 3.17, caso contrário, δ_j será dado pela Equação 3.20.

O principal problema do *backpropagation* diz respeito à lentidão do algoritmo para superfícies mais complexas. Uma forma de minimizar este problema é considerar efeitos de segunda ordem para o gradiente descendente. Não é raro convergir para mínimos locais. Mínimos locais são pontos na superfície de erro que apresentam uma solução estável, embora não seja a saída correta. Algumas técnicas são utilizadas tanto para acelerar o algoritmo *backpropagation* quanto para reduzir a incidência de mínimos locais: utilizar taxa de aprendizado decrescente; adicionar nós intermediários; utilizar um termo *momentum*; e adicionar ruído aos dados.

Entre as várias técnicas utilizadas para acelerar o processo de treinamento e evitar mínimos locais, a adição de um termo *momentum* (RUMELHART & MCCLELLAND, 1986) é uma das mais frequentes. Sua grande utilização é influenciada por ser simples e efetiva. O termo *momentum* é representado pela Equação 3.23:

$$\psi = \alpha(w_{ji}(t) - w_{ji}(t-1)) \quad \textbf{(3.23)}$$

Assim, a fórmula completa de ajuste de pesos utilizando o termo momentum e a taxa de aprendizado η passa então a ser:

$$w_{ji}(t+1) = w_{ji}(t) + \eta\delta_j(t)x_i(t) + \alpha(w_{ij}(t) - w_{ij}(t-1)) \quad \textbf{(3.24)}$$

A inclusão do termo *momentum* na fórmula de ajuste dos pesos aumenta a velocidade de aprendizado (aceleração), reduzindo o perigo de instabilidade. O termo momentum pode acelerar o treinamento em regiões muito planas da superfície de erro. Além disso, ele suprime a oscilação de pesos em vales e ravinas.

Capítulo IV – Predição de Estruturas Secundárias de Proteínas

Neste capítulo será realizada uma revisão na literatura da área de Predição de Estruturas Secundárias de Proteínas, com ênfase em soluções através de Redes Neurais. Serão apresentados conceitos básicos da área de Proteômica: química de proteínas, ângulos e conformações de proteínas e classificação de proteínas com base na sequência. O esclarecimento a respeito destes assuntos se faz necessário para o entendimento do restante do trabalho. Por fim, se realiza um levantamento cronológico dos trabalhos relevantes na área de predição de estruturas secundárias de proteínas utilizando Redes Neurais Artificiais, para posterior comparação com os resultados alcançados nesta pesquisa.

4.1 Introdução

Genoma é o conjunto completo de informações necessárias para o desenvolvimento de um organismo. Esta informação encontra-se armazenada nos cromossomos, que são constituídos essencialmente de ácido desoxirribonucleico (*DNA*). A cadeia de *DNA* contém milhares de genes, a partir dos quais são fabricadas todas as proteínas de um organismo.

As proteínas por sua vez têm papel essencial no metabolismo, participando praticamente de todas as atividades celulares. Existem diferentes tipos de proteínas, as quais exercem funções fundamentais nas células, tais como: suporte de filamentos, catálise bioquímica, regulação do volume celular, imunização, entre outras (NELSON & COX, 2000).

A Figura 4.1 apresenta o fluxo unidimensional da informação genética, que é: "o *DNA* atua como modelo para se replicar, ele também é transcrito em ácido ribonucleico (*RNA*) e este por sua vez, é convertido em proteína". Os projetos genoma focalizam a análise do DNA, estudos de transcriptoma produzem análise do conteúdo de *RNA*

mensageiro de uma célula e os projetos de proteoma geram a análise de conteúdo proteico.

Figura 4.1: Linearidade da Informação Genética

Os ácidos nucleicos e proteínas são moléculas formadas pela ligação de várias unidades semelhantes, denominadas monômeros. No *DNA* os monômeros são **nucleotídeos**, também chamados de bases. Estes são simbolizados por *A* (adenina), *C* (citosina), *G* (guanina) e *T* (timina). Nas proteínas os monômeros são os **aminoácidos**. Um total de 20 aminoácidos ocorre naturalmente nas proteínas, sendo cada um deles representado simbolicamente por uma letra: {*A*, *R*, *D*, *N*, *C*, *E*, *Q*, *G*, *H*, *I*, *L*, *K*, *M*, *F*, *P*, *S*, *T*, *W*, *Y*, *V*} (SETUBAL & MEIDANIS, 1997). A Tabela 4.1 apresenta os 20 aminoácidos juntamente com algumas características pertinentes. A representação de uma proteína através de sua sequência de aminoácidos é denominada estrutura primária, a qual será explanada com mais detalhes posteriormente. A determinação da sequência exata de monômeros que compõem tais moléculas é denominada sequenciamento.

Para entender a função de todos os genes em um organismo, é necessário conhecer não só quais genes são expressos, quando e onde, mas também quais são os produtos da expressão e em que condições esses produtos (proteínas) são sintetizados em certos tecidos. A Proteômica tenta descrever o conjunto completo de proteínas, o produto da expressão do genoma (JAMES, 1997).

Existe uma variedade de problemas em aberto para a comunidade científica em Biologia Computacional e na área de Proteômica, especificamente, existem alguns problemas que apesar de já estar sendo foco de pesquisa, ainda levará algum tempo para se obter resultados satisfatórios. Pode-se citar:

a) Análise e alinhamento de sequência de proteínas;

b) Predição da estrutura proteica;

c) Análise das propriedades da estrutura proteica; e

d) Alinhamento e comparação de estruturas proteicas.

Tabela 4.1: Tipos de aminoácidos e suas características

Nome	Símbolo	Massa	Ocorr. (%)	Natureza Química
Alanina	*A, Ala*	71,079	7,49	Hidrofóbico
Arginina	*R, Arg*	156,188	5,22	Básico
Asparagina	*N, Asn*	114,104	4,53	Polar
Ácido Aspártico	*D, Asp*	115,089	5,22	Ácido
Cisteína	*C, Cys*	103,145	1,83	Hidrofóbico
Glutamina	*Q, Gln*	128,131	4,11	Polar
Ácido Glutâmico	*E, Glu*	129,116	6,26	Ácido
Glicina	*G, Cly*	57,052	7,11	Hidrofóbico
Histidina	*H, His*	137,141	2,24	Básico
Isoleucina	*I, Ile*	113,160	5,45	Hidrofóbico
Leucina	*L, Leu*	113,160	9,06	Hidrofóbico
Lisina	*K, Lys*	128,170	5,82	Básico
Metionina	*M, Met*	131,199	2,27	Hidrofóbico
Fenilanina	*F, Phe*	147,177	3,91	Hidrofóbico
Prolina	*P, Pro*	97,117	5,12	Levemente Polar
Serina	*S, Ser*	87,078	7,34	Contém OH
Treonina	*T, Thr*	101,105	5,96	Contém OH
Triptofano	*W, Trp*	186,213	1,33	Hidrofóbico
Tirosina	*Y, Tyr*	163,176	3,25	Contém OH
Valina	*V, Val*	99,133	6,48	Hidrofóbico

4.2 Química das Proteínas

Proteínas são macromoléculas que têm funções específicas dentro do organismo. Elas podem ter caráter estrutural (como a queratina presente nos cabelos e o colágeno presente nos tendões e cartilagens) ou podem estar ligadas a determinadas atividades, como é caso dos hormônios (como a insulina) e das enzimas (como aquelas encontradas no estômago e relacionadas à digestão de alimentos).

Uma proteína é formada por unidades conhecidas como aminoácidos. Essas unidades ligam-se linearmente resultando uma cadeia conhecida como polipeptídio. Um aminoácido é composto por um carbono central (C_α), um hidrogênio (H), um grupo amino (H_2N), um grupo carboxila ($COOH$) e uma cadeia lateral R, conforme Figura 4.2.

A sequência linear de aminoácidos representa a estrutura primária das proteínas. Essas moléculas dobram-se e empacotam-se até um quarto nível, formando diferentes formas tridimensionais, que estão diretamente associadas à função bioquímica da proteína. As dobras se fazem em ângulos variados, decorrentes das ligações peptídicas. A existência de 20 diferentes aminoácidos propicia uma grande e complexa variedade

de formas irregulares, que determinam a sua ligação com outras moléculas. Cada proteína, com a sua forma, liga-se a tipos específicos de moléculas, ou seja, àquelas que possuem uma forma complementar a da proteína.

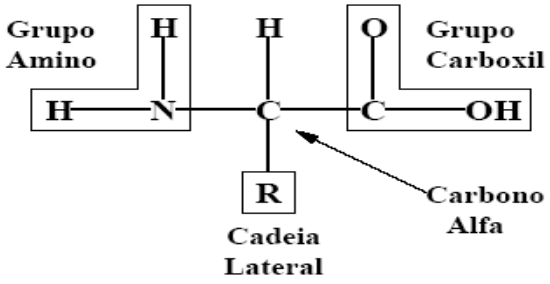

Figura 4.2: Estrutura química geral dos aminoácidos

Na determinação da função de uma proteína é importante conhecer quais as suas possíveis formas de apresentação objetivando determinar qual forma de análise é mais adequada. Como apresentado na Figura 4.3, as proteínas podem ser consideradas em quatro níveis de arquitetura: estrutura primária, estrutura secundária, estrutura terciária e estrutura quaternária.

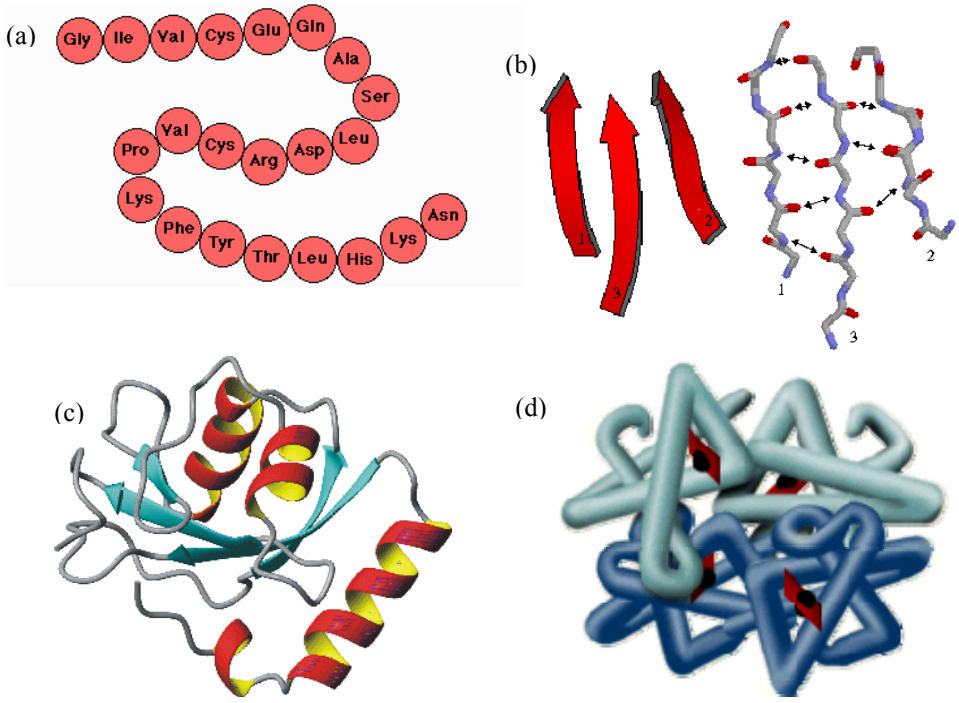

Figura 4.3: Os quatro níveis de representação de uma proteína

4.2.1 Estrutura Primária

Na estrutura primária, brevemente explanada anteriormente, as proteínas são representadas por subunidades monoméricas denominadas aminoácidos. São vinte os aminoácidos que ocorrem naturalmente nas diferentes formas de vida. A diferença entre eles é determinada por sua cadeia lateral ou grupo *R*. Os grupos *R* influenciam, por exemplo, na solubilidade do aminoácido em água e possuem diferentes estruturas, tamanhos e valência.

Ligações covalentes, denominadas peptídicas, unem uma quantidade de aminoácidos durante a síntese de proteínas, também denominadas cadeias polipeptídicas. Neste processo, há perda de uma molécula de água formada por um hidrogênio (*H*) do grupo amino com a hidroxila (*OH*) do grupo carboxila. O grupo amino do primeiro aminoácido na cadeia polipeptídica e o grupo carboxila do último aminoácido permanecem intactos. O resíduo que está localizado na extremidade que exibe o grupo amino é denominado amino-terminal ou *N*-terminal. Na outra extremidade, o resíduo que exibe o grupo carboxila é denominado carboxila-terminal ou *C*-terminal. A estrutura primária de uma proteína é convencionalmente lida da extremidade *N*-terminal para a *C*-terminal, conforme Figura 4.4. O arranjo recorrente de resíduos no espaço é denominado estrutura secundária, conforme Figura 4.5.

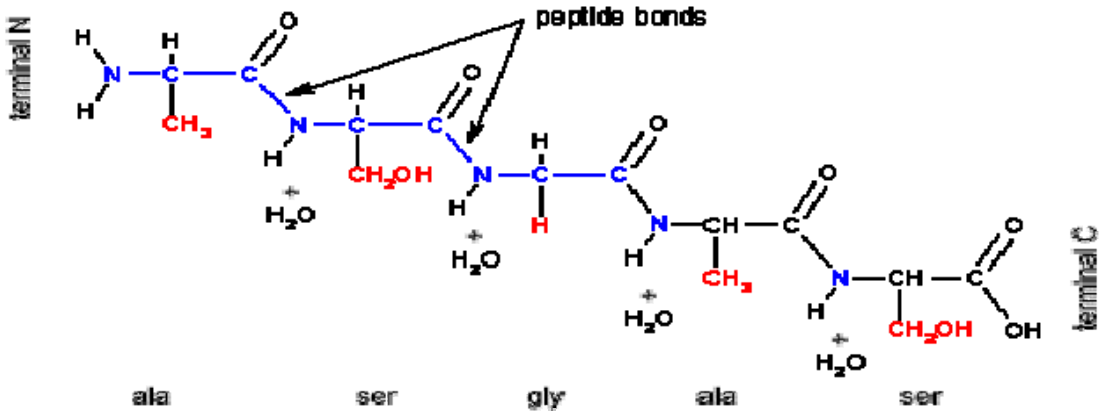

Figura 4.4: Estrutura primária de uma proteína

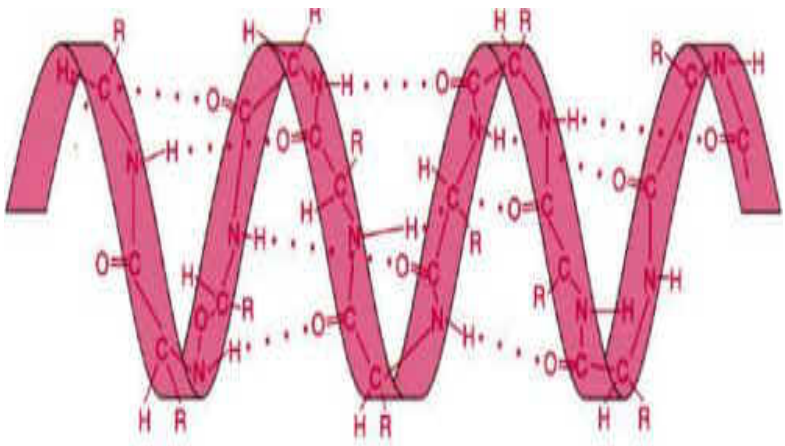

Figura 4.5: Estrutura secundária de uma proteína

4.2.2 Estrutura Secundária

Os elementos regulares da estrutura secundária podem ser distribuídos em três classes: *α*-hélices, folhas-*β* e *coils*.

As *α*-hélices são os elementos clássicos da estrutura da proteína e são estabilizadas por pontes de hidrogênio paralelas a seu eixo que ocorrem no interior do *backbone* de uma única cadeia polipeptídica. Contando-se a partir da extremidade *N*-terminal, o grupo *C=O* de cada resíduo de aminoácido está ligado por pontes de hidrogênio ao grupo *N-H* na sequência linear mantida por ligações covalentes (CAMPBELL, 2000). Assim, a conformação helicoidal permite um arranjo linear dos átomos envolvidos nas pontes de hidrogênio. Uma *α*-hélice (Figura 4.6) é um dos principais elementos de estrutura secundária, consistindo de uma hélice simples que pode ser formada de aproximadamente 10 a 40 resíduos consecutivos. Existem outros tipos de hélices, porém as *α*-hélices são energeticamente mais favoráveis.

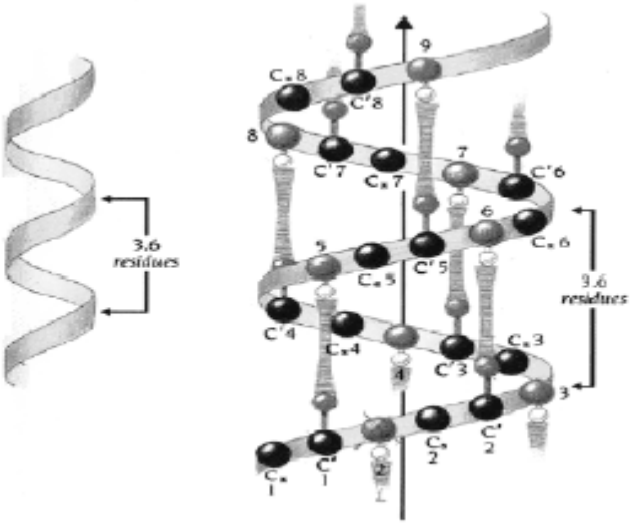

Figura 4.6: Representações esquemáticas de uma Alpha-Hélice

As folhas-*β*, por sua vez, são formadas pela combinação de várias regiões não contíguas, denominadas *β-strands*. Tais elementos são formados por 5 a 10 resíduos adjacentes e são ligados a outros *β-strands* através de pontes de hidrogênio, como pode

ser visualizado na Figura 4.7. As folhas-β diferem do arranjo das α-hélices, devido ao *backbone* peptídico das folhas-β estar quase estendido. Nessas folhas, as pontes de hidrogênio podem ser formadas entre diferentes partes de uma mesma cadeia dobrada sobre si mesma (pontes intracadeia) ou entre diferentes cadeias (pontes intercadeia). Se as cadeias peptídicas estendem-se numa mesma direção, isto é, se todas estiverem alinhadas em suas extremidades *N*-terminais e *C*-terminais, será formada uma folha pregueada paralela.

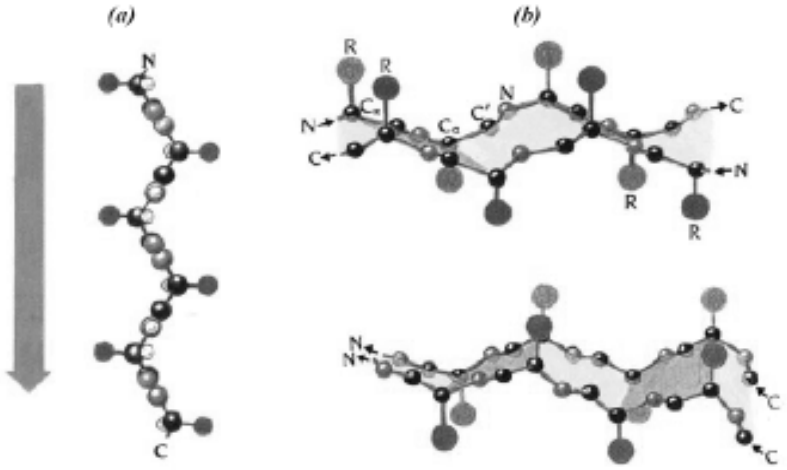

Figura 4.7: Beta-Folhas e as ligações entre os Beta-*strands*

A classe *coil* abrange os demais arranjos recorrentes que não foram incluídos nas duas primeiras classes, tais como hélices que não são energeticamente favoráveis e segmentos irregulares como os *loops*, que podem assumir formas e tamanhos diversos. *Coils* são estruturas irregulares com a não repetição dos ângulos de torção do *backbone* e, frequentemente, têm uma ligação de hidrogênio (CHANDONIA & KARPLUS, 1999).

4.2.3 Outras Estruturas

Essencialmente, a estrutura terciária de uma proteína consiste da conformação tridimensional dos elementos de estrutura secundária em uma única cadeia polipeptídica. Também neste arranjo é possível identificar subestruturas recorrentes. Proteínas com várias cadeias polipeptídicas têm ainda um nível adicional de arquitetura: a estrutura quaternária. Esta especifica os relacionamentos espaciais entre os polipeptídios ou subunidades destes.

Existem ainda alguns níveis intermediários de estruturas: as estruturas supersecundárias e os domínios. As estruturas supersecundárias, ou *motifs* (motivos), são conjuntos estáveis de elementos de estrutura secundária, os quais formam arranjos particulares através de interações entre as cadeias laterais. Podendo estar ou não associados a uma função, os *motifs* podem ocorrer várias vezes em uma mesma proteína ou em proteínas diferentes. Um exemplo de estrutura supersecundária é o assim chamado *β-hairpin* o qual consiste de dois *strands* antiparalelos unidos por um *loop*. Os domínios são regiões compactas que podem compreender de 40 a 400 aminoácidos formando uma unidade estrutural distinta na região conservada da proteína (o *core*). Tais estruturas estão necessariamente associadas a uma função e, alguns arranjos são energeticamente mais favoráveis do que outros. É importante observar que na literatura da Biologia Molecular estes termos podem ter outras conotações, sendo associados, por exemplo, a elementos característicos de sequências.

4.2.4 Ângulos e a Conformação das Proteínas

As ligações peptídicas possuem propriedades importantes para a estrutura de uma proteína. A ligação do carbono ao nitrogênio, na ligação peptídica, possui um caráter de ligação dupla parcial (ressonante), o que implica que há uma restrição, a que o ângulo é fixado em torno de 180° para cada aminoácido. Há uma liberdade rotacional nas ligações que conectam o carbono alfa (C_α) às duas unidades planares adjacentes formando dois ângulos chamados φ e ϕ, que conferem à cadeia principal da proteína ou esqueleto (*backbone*), sua liberdade de conformação, como apresentado na Figura 4.8.

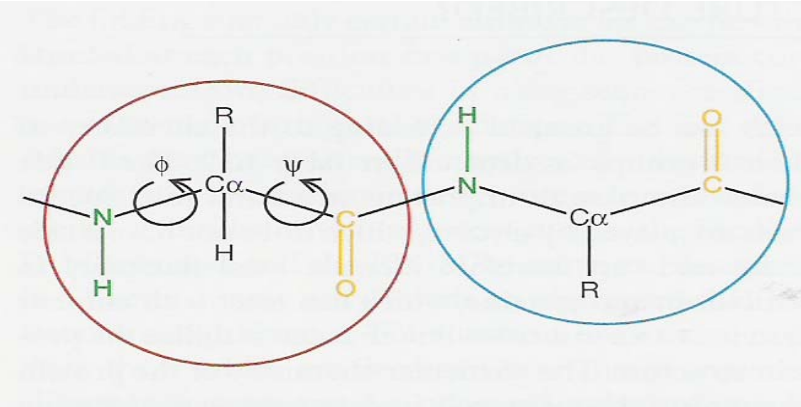

Figura 4.8: Estrutura de dois aminoácidos em uma cadeia de polipeptídeos

Dessa forma pode-se descrever a estrutura do *backbone* da proteína através dos valores dos ângulos φ e ϕ e pode-se, assim, descrever a conformação de um resíduo na proteína a partir de um ponto em um mapa com as coordenadas desses ângulos, denominado mapa de Ramachandran, conforme Figura 4.9. Nesse mapa, os valores de φ e ϕ variam entre –180° e +180°. É possível, conhecendo o gráfico de Ramachandran, descrever quais as estruturas secundárias de uma proteína a partir do valor de cada par de ângulos φ e ϕ associado a cada ligação peptídica da cadeia (*α-hélices, β-folhas e coils*).

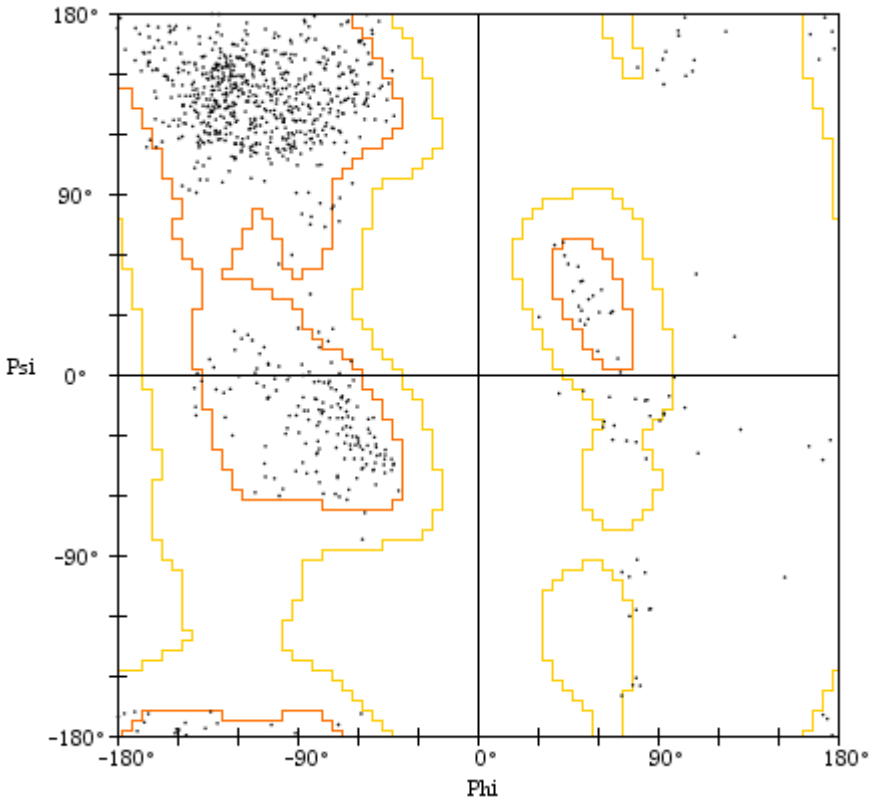

Figura 4.9: Exemplo de Mapa de Ramachandran

Proteínas possuem um alto grau de liberdade de conformação devido aos valores que os ângulos φ, e ϕ podem assumir e isso pode introduzir uma complexidade na busca de conformação para essas macromoléculas e na predição de suas estruturas secundárias e terciárias (NELSON & COX, 2000).

A função proteica está diretamente relacionada com a sua estrutura nativa. O processo a partir do qual a proteína parte da conformação aleatória e alcança sua estrutura nativa é conhecido como *protein folding* (enovelamento da proteína). Isto é, além da estrutura correspondente ao estado nativo, as proteínas também podem possuir

várias estruturas intermediárias (*random coil*) e este processo no qual a proteína sai do estado aleatório e atinge o seu estado nativo é, atualmente, um dos problemas de maior relevância na comunidade científica.

Ao sair de uma conformação aleatória, as proteínas alcançam sua estrutura nativa, conhecida como "enovelada". Com o estudo deste processo, pretende-se descobrir quais as propriedades da proteína que levam sua cadeia a adotar uma estrutura única e estável e como sua sequência primária de aminoácidos se relaciona com essas propriedades.

4.3 Predição de Estruturas com Base na Sequência

É possível, através de alguns métodos, predizer aspectos mais simplificados da estrutura terciária (tridimensional) de uma proteína a partir de sua sequência (estrutura primária) (ROST, 1996) (ROST, 1998). Um aspecto mais simplificado seria a estrutura em uma dimensão que são os contatos dos resíduos adjacentes na sequência (predição da estrutura secundária da proteína). Alguns métodos utilizam redes neurais artificiais e algoritmos genéticos para predizer estruturas secundárias e terciárias a partir de determinadas sequências primárias de aminoácidos (ROST, 1996) (ROST, 1998) (SCOTT, 2003).

A análise da sequência proteica é baseada em parte na compreensão das propriedades físico-químicas dos componentes da cadeia da proteína e, em parte, no conhecimento da frequência de determinados aminoácidos em posições específicas nas estruturas e subestruturas proteicas. Apesar das ferramentas de análise de sequências de proteína operarem em dados de sequências unidimensionais, elas contêm suposições implícitas sobre como as características estruturais se relacionam aos dados da sequência, como por exemplo, ao se utilizar informações como grau de hidrofobicidade dos aminoácidos.

A predição da estrutura secundária é considerada, em geral, o primeiro passo na predição da estrutura de uma proteína. Um método de predição de estruturas secundárias de proteínas tem o objetivo de classificar resíduos adjacentes em padrões do tipo *H* (α-hélices), *F* (folhas-β) e *C* (*coil* ou fita aleatória).

Um ponto importante dos métodos de predição de estrutura secundária é o fato de que segmentos de resíduos consecutivos possuem uma preferência por certos estados de estrutura secundária. Então o problema de predição de estrutura torna-se um

problema clássico de classificações de padrões, que é tratável por algoritmos de reconhecimento de padrões, por exemplo, redes neurais artificiais.

Os algoritmos destes métodos para predição podem ser agrupados da seguinte forma:

- Utilizam informações estáticas;
- Utilizam propriedades físico-químicas;
- Fazem uso de padrões de sequências;
- Utilizam redes neurais artificiais; e
- Exploram a conservação evolucionária.

Existem algoritmos que combinam ideias, como algoritmos que utilizam redes neurais artificiais e informações evolucionárias (SCOTT, 2003). Os métodos de predição podem ser classificados em três categorias: primeira, segunda e terceira gerações (ROST & SANDER, 1993), (ROST, 1996) (ROST, 1997) (ROST, 1998) (ROST & SANDER, 2000), (BALDI; BRUNAK *et al*., 1999).

Os métodos da primeira geração são baseados em propriedades físico-químicas (propensão de certos resíduos formarem α-hélices e folhas-β), regras e estatísticas de resíduos isolados, ou seja, não utilizavam informações dos resíduos adjacentes.

Os métodos da segunda geração incorporam bases de dados maiores e estatísticas, baseadas em segmentos de resíduos adjacentes, tipicamente entre 11 e 21 resíduos. Dessa forma, eles consideram a influência dos resíduos adjacentes sobre o resíduo para a qual a estrutura secundária foi predita (ROST & SANDER, 1993).

Os métodos de terceira geração são superiores aos métodos da segunda geração, porque procuram tratar esses problemas simultaneamente. Possuem como principais características: o fato de utilizar informações evolucionárias obtidas a partir do alinhamento múltiplo de sequências, realizando uma associação explícita entre resíduos de duas ou mais sequências; o uso de múltiplos níveis de computação e o treinamento balanceado das redes neurais artificiais (ROST & SANDER, 2000).

Os métodos de terceira geração são os mais utilizados na predição de estruturas secundárias de proteínas através de redes neurais artificiais, possuindo uma acurácia consideravelmente maior em relação aos outros métodos. A identificação com sucesso da estrutura secundária é pré-requisito para uma predição bem sucedida de uma parte de todos os contatos inter-resíduos. Contudo, contatos que foram preditos a partir de associação de estrutura secundária são de curto alcance, isto é, entre resíduos próximos

na sequência. É necessário predizer, também, os contatos de longo alcance, entre resíduos distantes na sequência.

4.4 Predição de Estruturas Secundárias utilizando Redes Neurais

Entre as possíveis aplicações de *RNAs* no estudo de estruturas de moléculas como proteínas estão: predição de estruturas secundárias através de reconhecimento de padrões; predição de estruturas terciárias de proteínas através de otimização de uma função potencial de energia; predição de possíveis sequências de aminoácidos para uma dada proteína, de forma a obter as conformações de mais baixa energia; predição de sequências de aminoácidos que levem a uma estrutura secundária ou terciária desejada, entre outras.

Pode-se afirmar que a predição de estrutura secundária é um problema de classificação. Este consiste em classificar cada resíduo de uma sequência de aminoácidos em uma das subestruturas recorrentes na conformação *3D* de uma proteína, as quais podem ser grupadas em: α-hélices, β-folhas ou *coils*. A utilização de redes neurais para efetuar tal classificação tem-se revelado uma abordagem eficiente, presente nos melhores preditores disponíveis atualmente (POLLASTRI, PRZYBYLSKI *et al*., 2002), (JONES, 1999).

4.4.1 Trabalho de Qian & Sejnowski (1988)

Um exemplo clássico da aplicação desta técnica de Inteligência Artificial para tal problema é o trabalho de Qian e Sejnowski (QIAN & SEJNOWSKI, 1988), no qual um estudo preliminar para escolha da melhor configuração de uma rede neural para predição de estrutura secundária é apresentado. Através deste, pode-se compreender os princípios básicos que são observados no desenvolvimento de um preditor. As sequências de proteínas são percorridas por janelas, de diferentes tamanhos, que se sobrepõem, sendo o treinamento e a predição realizados em relação ao elemento central de tais janelas (Figura 4.10).

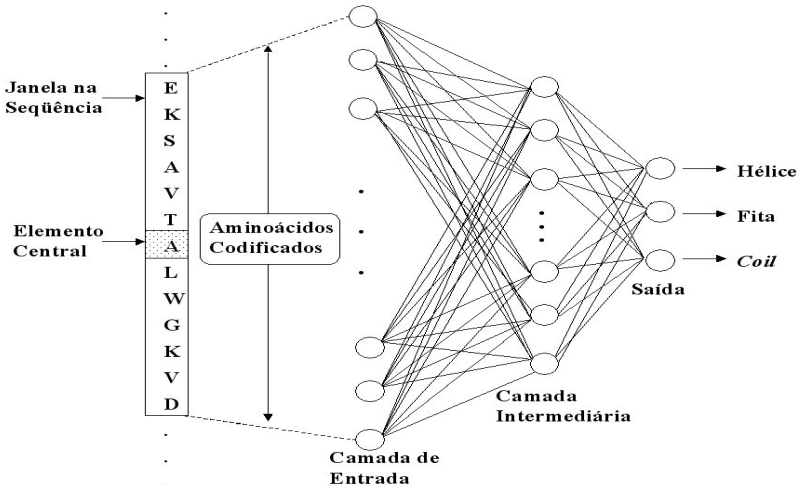

Figura 4.10: Estrutura da Rede Neural *MLP* proposta por Qian & Sejnowski (1988)

Os aminoácidos são codificados ortogonalmente e as interseções das janelas com as regiões *N*-terminal e *C*-terminal são representadas por * correspondendo ao zero na representação ortogonal. O formato escolhido para simbolizar cada um dos 20 aminoácidos mais os *gaps* leva a uma codificação de 21 números para cada resíduo. A mesma representação unária é aplicada às classes alvo.

As redes utilizadas por Qian e Sejnowski são *Perceptrons* Multi-Camadas (*MLPs*) com uma camada intermediária. Foram realizados experimentos variando os tamanhos de janela e o número de nós na camada intermediária, sendo o número de saídas sempre igual a três: um nó para cada classe. As redes foram treinadas com o tradicional algoritmo *backpropagation* utilizando para isto um subconjunto de um banco de dados com 106 sequências de proteínas globulares dissimilares. Dois subconjuntos do banco, disjuntos do empregado para treinamento, foram utilizados para testes: um com quinze sequências e outro com seis. O desempenho da rede é avaliado através de uma medida de desempenho bastante utilizada, denominada Q_3 (SCHULZ & SCHIRMER, 1979). Tal medida fornece a porcentagem de resíduos classificados corretamente:

$$Q_3 = \frac{número\_de\_resíduos\_classificados\_corretamente}{número\_total\_de\_resíduos} \text{x}100 \qquad \textbf{(4.1)}$$

O melhor desempenho Q_3 reportado por Qian e Sejnowski neste experimento foi de 64.3%, utilizando uma janela de tamanho 13, e 40 nós na camada escondida. Este

resultado foi superior a qualquer outro obtido através de métodos estatísticos, mais utilizados na época.

A partir do sucesso alcançado com esta abordagem, diferentes arquiteturas, algoritmos e tratamento de entrada de dados foram experimentados (BALDI & BRUNAK, 2001), (ROST, 2001).

4.4.2 Trabalho de Holley & Karplus (1991)

Um dos primeiros trabalhos que utilizou redes neurais na predição de estruturas secundárias de proteínas a partir de sequências primárias foi dos pesquisadores Holley e Karplus (HOLLEY & KARPLUS, 1991). Neste trabalho, Holley & Karplus utilizaram redes neurais para associar as proteínas a uma das quatro classes (todas, ou seja, todas as classes em conjunto único; α-hélices e folhas-β; todas folhas-β; todas α-hélices).

Eles utilizaram uma *RNA* do tipo *MLP* e codificaram os dados de entrada da rede em janelas de resíduos adjacentes. Para cada resíduo, há 21 (vinte e uma) entradas binárias, ou seja, cada aminoácido foi codificado em 21 (vinte e uma) unidades. Sendo 20 (vinte) unidades representando os aminoácidos e 1 (uma) unidade representando o heteroátomo (*null*). A *RNA* utilizada possuía uma camada intermediária com duas unidades e uma camada de saída com duas unidades.

Holley e Karplus utilizaram um conjunto de dados de 48 proteínas para treinar e 14 proteínas para teste. Eles testaram vários tamanhos de janela e a que mostrou os melhores resultados foi a janela de 17 (dezessete) resíduos. Também foram testadas *RNAs* com diferentes tamanhos de camadas intermediárias (variando de 2 a 20). Apesar da rede com camada intermediária contendo 20 (vinte) unidades ter apresentado o melhor resultado para o conjunto de treinamento, a rede com a camada intermediária com 2 (duas) unidades apresentou melhor resultado para o conjunto de teste. O desempenho obtido para o treinamento foi de 68,5% e para o de teste foi de 63,2%.

4.4.3 Trabalho de Rost & Sander (1993 e 1994)

Uma melhoria na acurácia das predições foi obtida através da introdução de mais informação biológica para as redes, mais especificamente informações evolucionárias, por meio do uso de perfis das sequências de proteínas como dados de entrada. Ao contrário das sequências que fornecem apenas uma informação local, os perfis refinam a busca no banco de dados identificando relações mais distantes, permitindo assim uma maior precisão na predição.

Um exemplo deste ganho pode ser visto no trabalho de Rost e Sander (ROST & SANDER, 1993) o qual resultou no conhecido preditor *PHD* (ROST & SANDER, 1994). Rost e Sander aplicaram a mesma estrutura que gerou os melhores resultados na proposta de Qian e Sejnowski, porém, utilizando como dados de entrada os perfis de frequência. Tais perfis são armazenados em matrizes, cujas entradas são calculadas com base na frequência de cada um dos 20 aminoácidos nas colunas de alinhamentos múltiplos das sequências usados no banco *HSSP*.

Em 1993, Rost e Sander (ROST & SANDER, 1993) utilizaram uma *RNA* do tipo *MLP*, com duas camadas intermediárias, sobre uma base de dados não redundante de 130 proteínas. O diferencial deste trabalho foi o uso de informações evolucionárias (informações de segmentos da estrutura primária que se mantém conservada) obtidas por alinhamentos múltiplos de sequências que são utilizadas como entrada ao invés de sequências simples. Esse foi um dos primeiros trabalhos a incluir informação evolucionária para auxiliar as redes neurais na predição de estruturas e apresentou um aumento de 6 a 8% na capacidade de predição. A combinação de três níveis de redes resultou em taxa de acerto de 70,8%. O trabalho utilizou três redes neurais de forma que a primeira rede utiliza 20 unidades de entrada para cada resíduo e uma janela de 13 resíduos. A segunda rede leva em consideração a correlação entre os segmentos. Nessa rede, a entrada é uma janela de 17 resíduos adjacentes e possui como entrada, a saída da primeira rede. Na terceira rede é aplicado um júri na saída de diferentes redes, computando a média aritmética para 8 redes neurais diferentes.

No trabalho de 1994 (ROST & SANDER, 1994) cada aminoácido é representado por 21 números: 20 linhas da matriz do perfil e o número 1 indicando a sobreposição com os *N*- e *C*-terminais. Além do formato de entrada, outro ponto diferencial foi introduzido por Rost e Sander: o treinamento balanceado. Em proteínas globulares, o número de resíduos *coils* é significantemente alto ao comparamos com o número de hélices e folhas. Para o banco de dados utilizado por Rost e Sander a distribuição é de 32% de α-hélices, 21% de folhas-β e 47% de *coils*. Isto causa um valor alto para predição de *coils* e baixo para as folhas-β, os quais aparecem em menor proporção. Utilizando o treinamento balanceado cada classe de estrutura secundária é utilizada em igual proporção, melhorando assim a predição para a classe folhas-β.

As saídas da rede que realiza a predição da classe baseada na sequência, rotulada sequência-estrutura, são agrupadas de modo a corresponder a janelas de tamanho 17 nas sequências e filtradas por uma segunda rede, denominada estrutura-estrutura. O

treinamento estrutural permite o reconhecimento do tamanho da distribuição de hélices e de folhas. Um conjunto composto de 12 destas redes é utilizado para combinar as predições assim obtidas através da maioria simples, resultando em um desempenho Q_3 de 71.9%, ou seja, cerca de 7% superior ao melhor resultado reportado em (QIAN & SEJNOWSKI, 1988).

4.4.4 Trabalho de Chandonia & Karplus (1996)

Chandonia e Karplus (CHANDONIA & KARPLUS, 1996) aplicaram duas redes neurais, denominadas primária e secundária, e foi utilizado um conjunto de 681 proteínas com estruturas disponíveis no *PDB* (*Protein Data Bank*). O desempenho Q_3 atingiu 67.0% para este experimento.

A rede primária utilizada por Chandonia e Karplus foi similar à rede da pesquisa descrita no trabalho de Holley & Karplus (HOLLEY & KARPLUS, 1991), com 21 unidades de entrada que representam o tipo de aminoácido, sendo o último representando o fim da cadeia. A saída desta rede possui 2 (duas) unidades que correspondem à estrutura secundária, α-hélices e folhas-β.

A rede neural secundária é utilizada para refinar os resultados produzidos pela primeira rede, classificando as proteínas em classes: todas (conjunto que possui todas as classes: α-hélices, folhas-β e *coils*); todas α-hélices (conjunto que possui somente proteínas com características α-hélices); e todas folhas-β (conjunto que possui somente proteínas com características folhas-β). Nessa segunda rede, os dados de entrada são os resultados da primeira rede, α-hélices e folhas-β, para cada resíduo.

4.4.5 Trabalho de Riis & Krogh (1996)

Outro aspecto a ser considerado pelos classificadores é o tamanho dos dados de entrada. Através da representação ortogonal, ou mesmo a utilização de perfis, cerca de 20 nós são necessários na camada de entrada para a codificação de cada aminoácido de uma janela. Uma interessante abordagem para tal problema foi proposta por Riis e Krogh (RIIS & KROGH, 1996). As topologias das redes utilizadas pelo preditor foram projetadas de forma a reduzir o número de parâmetros, evitando assim *overfitting*.

Para treinamento e teste das redes, Riis e Krogh utilizaram o banco de dados *RS126*, desenvolvido por Rost e Sander (ROST & SANDER, 1994). Empregado por diversos preditores, o *RS126* consiste de 126 proteínas globulares não-homólogas, retiradas do *HSSP* (SANDER & SCHNEIDER, 1994). A medida de homologia utilizada

para seleção das sequências foi a similaridade entre pares, cujo percentual de identidade aceito era de no máximo 25% para sequências com mais de 80 resíduos.

O método de avaliação utilizado foi uma variação econômica da técnica do tipo *jack-knife*, chamada *seven-fold cross-validation*. Nesta, o conjunto de sequências é dividido em sete partes de tamanhos aproximadamente iguais. Seis partes são utilizadas para treinamento e uma para teste, sendo este processo repetido ciclicamente.

A codificação local dos aminoácidos é feita aplicando um caso particular do método *weight sharing*. Neste, as próprias redes escolhem a melhor representação para os aminoácidos, os quais são inicialmente codificados ortogonalmente. Os 20 valores associados a cada aminoácido são conectados a M unidades na camada escondida através de 20 x M pesos. Para forçar que cada aminoácido tenha a mesma codificação, o algoritmo *backpropagation* é modificado de forma que na atualização os pesos compartilhem os mesmos valores.

Após a codificação, janelas de 15 aminoácidos são informadas a redes desenvolvidas para predizer cada uma das três classes separadamente. Tais redes contêm apenas uma saída e a decisão é arbitrada em um limite de 0.5. Se a saída é maior do que este valor, então a entrada correspondente é classificada como sendo da estrutura em consideração. Para predizer cada tipo de estrutura secundária uma combinação de cinco dessas redes é utilizada, cada uma com diferentes números de nós na camada escondida. Com a codificação empregada, o número de pesos ajustáveis de todas as redes juntas é menor do que 600. As saídas de cada grupo de cinco redes são combinadas por outra rede estrutura-estrutura, cuja predição do resíduo central é escolhida como a maior das três saídas, as quais são normalizadas com a função *Softmax* (BRIDLE, 2000). Com este arranjo, o desempenho Q_3 alcançado utilizando sequências de proteínas como entrada foi de 66.3%.

Adaptando o método para considerar os alinhamentos múltiplos das sequências o preditor de Riis e Krogh alcança uma taxa de acerto de 71.3%. Nesta abordagem, para evitar a perda de correlações entre os aminoácidos, as predições são feitas para sequências simples e então combinadas através de alinhamentos múltiplos. Neste método, para cada proteína, chamada proteína base, um conjunto de proteínas homólogas é encontrado. Então, a predição de estrutura secundária é feita para cada proteína no conjunto, incluindo a proteína base, utilizando o procedimento descrito anteriormente. As proteínas do conjunto são alinhadas e um peso é associado a cada uma delas. Em seguida, uma predição consenso é encontrada para cada coluna do

alinhamento, baseada nas predições realizadas para cada aminoácido na coluna. Este consenso pode ser obtido por um dos métodos: *weighted average* e *weighted majority*. No primeiro método, as predições de cada classe são multiplicadas pelo peso da proteína e então são somadas. A maior das três somas determinará a predição para a coluna correspondente. No método *weighted majority*, a predição de cada aminoácido na coluna é escolhida como sendo a maior das três saídas. Os pesos para cada classe são somados e, então, a classe que obtiver a maior soma é escolhida como a predição do aminoácido daquela coluna.

4.4.6 Contribuições mais recentes

Os elementos explanados anteriormente podem ser encontrados em abordagens desenvolvidas mais recentemente (GUIMARÃES, MELO *et al*., 2003), (POLLASTRI, PRZYBYLSKI *et al*., 2002), (ROST & SANDER, 2000), (BALDI, BRUNAK *et al.*, 1999). No entanto, ainda há espaço para melhoramentos, variações e aplicações de novos métodos são propostos objetivando alcançar predições mais confiáveis.

Dentre as aplicações mais recentes, um aumento significativo no desempenho foi obtido através de uma modificação do tipo de entrada, introduzindo mais informação divergente, particularmente os perfis *PSI-Blast* (PRZYBYLSKI & ROST, 2002). Este tipo de entrada foi proposto pela primeira vez por Jones (JONES, 1999) no preditor *PSIPRED*. O programa *PSI-Blast* (ALTSCHUL, MADDEN *et al.*, 1997) pode encontrar homólogos distantes de sequências em um banco de dados. Os perfis *PSI-Blast* são gerados como parte do processo de busca do programa homônimo e, sua obtenção é mais rápida do que a construção do alinhamento múltiplo explicitamente.

A arquitetura do preditor *PSIPRED* consiste de uma rede sequência estrutura com uma janela de entrada de tamanho 15, cujos aminoácidos são representados através das linhas da matriz contendo o perfil *PSI-Blast* da sequência correspondente, com 75 nós na camada escondida e 3 na camada de saída. Nesta abordagem, saídas sucessivas, correspondentes a uma janela de tamanho 15 na sequência, também são filtradas por uma segunda rede, estrutura-estrutura, com 60 nós na camada escondida. O algoritmo utilizado para treinamento foi o *backpropagation* com um termo de *momentum* de 0.9 e taxa de aprendizagem de 0.005. Para avaliar o desempenho, 10% do conjunto de treinamento é reservado. Desta forma, o desempenho Q_3 obtido pelo preditor *PSIPRED* ficou entre 76.5% e 78.3%. Jones sugere em seu trabalho que a simples substituição da entrada por perfis *PSI-Blast* pode melhorar o desempenho de outros preditores (JONES,

1999). Em um estudo recente, Cuff e Barton (CUFF & BARTON, 2000) estimaram que o crescimento médio da taxa de acerto das predições com perfis *PSI-Blast* foi de 0.5% em relação a predições realizadas com alinhamentos múltiplos gerados com o programa *CLUSTALW* (THOMPSON, HIGGINS *et al.*, 1994).

Após o excelente desempenho alcançado pelo *PSIPRED*, a grande maioria dos preditores passou a utilizar perfis *PSI-Blast*, obtidos a partir de bancos e parâmetros diferentes, como entrada para as redes. Eles foram aplicados, por exemplo, no classificador *SSPro* 2.0 em conjunto com redes neurais recorrentes (POLLASTRI, PRZYBYLSKI *et al.*, 2002). No trabalho proposto por Pollastri, Przybylski *et al.*, os perfis *PSI-Blast* são obtidos através de um procedimento de quatro passos. Estes incluem a remoção das regiões de baixa complexidade e *coiled-colis*, a execução de três interações do programa *PSI-Blast* com *e-value* de 10^{-10}, a utilização do perfil produzido para alinhar as sequências com as de um banco não filtrado e, finalmente, o balanceamento e remoção de redundâncias do perfil. Onze redes neurais bidirecionais recorrentes (*BRNNs*) compõem o preditor. O contexto "esquerda-direita" é produzido através de duas redes neurais recorrentes: uma iniciando a predição pela extremidade *N*-terminal da sequência, e a outra pela extremidade *C*-terminal. O banco de dados utilizado para treinamento consiste de 1180 sequências não redundantes e a fase de testes foi realizada com três bancos de dados diferentes: *RS126* (ROST & SANDER, 1994), *EVA* (ROST & VA, 2001) e *CASP4* (CASP4, 2000). O desempenho Q_3 obtido com o *RS126*, discutido anteriormente, foi de 76.62%. Utilizando o banco de dados *EVA*, o qual consiste de 223 sequências de proteínas dissimilares do conjunto de treinamento, a taxa de acerto foi de 76%. Finalmente, o melhor desempenho, 77.8%, foi alcançado com o banco de dados *CASP4*, o qual consiste de 40 proteínas, porém é importante ressaltar que algumas delas apresentaram similaridade com o conjunto de treinamento.

Recentemente, Petersen, Lundegaard *et al.* (PETERSEN, LUNDEGAARD *et al.*, 2000) utilizaram perfis *PSI_Blast* para realizar uma predição em dois níveis. O banco de dados utilizado para treinamento e teste consiste de 1032 sequências, cada uma com pelo menos 30 resíduos, selecionadas do *PDB* (BERMAN, WESTBROOK *et al.*, 2000). Sequências similares e transmembranas foram removidas do banco de dados, juntamente com todas as sequências que apresentavam alta similaridade como aquelas presentes no *RS126*. Os perfis *PSI_Blast* foram gerados de forma a filtrar todos os

resíduos anotados como *RICH, COIL, REPEAT, HYDROFOBIC, SIGNAL* ou *TRANSMEMBRANE* (BRANDEN & TOOZE, 1999).

Além do perfil das sequências, seis nodos são usados na entrada: dois para a codificação da posição relativa do resíduo na sequência, dois para a codificação do tamanho relativo da sequência e dois para a codificação da frequência dos aminoácidos também na sequência. As redes envolvidas na predição foram treinadas com o tradicional algoritmo *backpropagation* e apresentam diferentes números de nós na camada de entrada e na camada intermediária. As oito arquiteturas utilizadas resultam da combinação de janelas de tamanhos 15, 17, 19 ou 21, com 50 ou 75 unidades na camada intermediária. Para a camada de saída uma técnica chamada expansão da saída é usada. Nesta, uma predição simultânea do resíduo central e sua vizinhança é feita. Especificamente na abordagem desenvolvida por Petersen, Lundegaard *et al.*, foram considerados os vizinhos imediatamente antes e após o resíduo central, resultando em nove unidades de saída (três para cada classe considerada: hélice, folha e *coil*). Uma rede estrutura-estrutura também é utilizada para filtrar as saídas da rede sequência-estrutura. Esta rede utiliza uma janela de tamanho 17 como entrada, possui 40 unidades na camada intermediária e três nodos de saída. Um método estatístico, chamado *balloting*, é empregado para combinar as 800 predições geradas pelo método utilizado, o *ten-fold cross-validation*.

O desempenho Q_3 alcançado em (PETERSEN, LUNDEGAARD *et al.*, 2000), combinando as 800 predições das redes segundo tal procedimento foi de 77.2%. Utilizando o *RS126* para teste o desempenho chega a 80.6%, valor considerado um limite a ser superado por qualquer outro trabalho.

Embora os resultados apresentados sejam relativamente próximos, compará-los não é uma tarefa fácil. Como pode ser observado, a utilização de diferentes bancos de dados e métodos de avaliação é uma prática comum no desenvolvimento de preditores e uma comparação nestas condições não é precisa (ROST, 2001), pois nada garante que um bom desempenho com um conjunto de dados possa também ser alcançado com outro conjunto de proteínas.

Tal problema foi abordado recentemente por Cuff e Barton (CUFF & BARTON, 2000). Neste trabalho foi realizado um estudo comparativo de quatro preditores de estrutura secundária utilizando os mesmos bancos de dados e procedimentos de avaliação para todos os classificadores estudados. Além do preditor *PHD* (ROST & SANDER, 1994), discutido anteriormente, três outros classificadores que utilizam

entrada de dados baseados em alinhamentos múltiplos foram analisados, a saber: o *DSC* (KING & STERNBERG, 1996), o *NNSSP* (SALAMOV & SOLOVYEV, 1995) e o *PREDATOR* (FRISHMAN & ARGOS, 1997).

O preditor *DSC* é fundamentado no método *GOR* (GARNIER, OSGUTHORPE *et al.*, 1978) o qual se baseia em Teoria da Informação para efetuar a predição em quatro estados conformacionais, cujos *scores* são corrigidos através de constantes de decisão que podem ser dadas pelo usuário. Além do alinhamento múltiplo de sequências, o *DSC* recebe como entrada o grau de hidrofobicidade e a posição relativa de cada aminoácido. O preditor *NNSSP* é baseado no algoritmo *nearest-neighbor* cuja predição do resíduo central da janela é realizada através de um segmento de teste pontuado através da similaridade com outros segmentos de estrutura conhecida. O *PREDATOR* por sua vez, utiliza métodos de alinhamentos de pares de sequências e informações sobre elementos de estrutura secundária, encontrados através de um procedimento baseado no algoritmo *nearest-neighbor*, além de incorporar informações estatísticas de interações não locais. Adicionalmente, Cuff e Barton (2000) avaliaram a combinação desses quatro métodos, chamada *CONSENSUS*. Tal combinação foi efetuada por meio de um algoritmo desenvolvido pelos próprios autores, o qual calcula a predição de cada resíduo através da votação dos quatro preditores e, em caso de impasse, o resultado reportado pelo preditor *PHD* é usado.

Para a fase de treinamento e teste das redes neurais foram utilizados dois bancos de dados: O *RS126*, desenvolvido por Rost e Sander (ROST & SANDER, 1994) e o *CB396*, um banco de dados desenvolvido por eles também. O *RS126* foi escolhido por ser um banco de dados bastante utilizado para treinamento e teste de diversos preditores conhecidos. Já o *CB396* foi desenvolvido de modo a garantir certo grau de dissimilaridade, medidas de forma apropriada, entre as proteínas representadas pelas sequências utilizadas para teste e treinamento. Cuff e Barton (2000) reforçam a importância da utilização de bancos de dados com esta característica como uma forma de evitar desempenhos que não correspondem à realidade. No banco de dados desenvolvido por eles, ao invés do percentual de identidade utilizado como critério de seleção no *RS126*, as sequências são escolhidas por meio de um algoritmo de comparação bastante sensível e de análise de *clusters* aplicados ao banco de dados *3Dee* (SIDDIQUI, DENGLER *et al.*, 2001). Segmentos com multi-domínios, sequências com resolução de raio *X* menor ou igual a 2.5 *angstrons*, bem como sequências similares

àquelas presentes no *RS126* são removidas. As 396 sequências resultantes deste processo compõem o *CB396*.

Dos quatro preditores avaliados por Cuff e Barton (2000) o que apresentou melhor desempenho individual foi o *PHD*, que alcançou um desempenho Q_3 de 73.5% para o banco de dados *RS126* e 71.9% quando avaliado sobre o banco *CB396*. Porém os melhores índices foram alcançados com o preditor *CONSENSUS*, resultante da combinação dos métodos avaliados. A taxa de acerto obtida com este foi de 74.8% para o *RS126* e 72.9% para o banco *CB396*.

Pode-se observar que, para obter melhores resultados, os recursos computacionais requeridos são cada vez maiores. Além da complexidade intrínseca a cada classificador individualmente, a maioria dos preditores combina algo como oito, onze, doze, ou mesmo oitocentas predições (BALDI, BRUNAK *et al.*, 1999), (POLLASTRI, PRZYBYLSKI *et al.*, 2002), (ROST & SANDER, 1993), (PETERSEN, LUNDEGAARD e*t al*, 2000), conforme explanado anteriormente. Em um esforço para modificar esta tendência, Guimarães, Melo *et al.* (GUIMARÃES, MELO *et al.*, 2003) desenvolveram o *GMC*, um preditor de estrutura secundária de arquitetura simples e cuja eficiência é comparável à de outros preditores avaliados sobre os estabelecidos bancos de dados *RS126* e *CB396*.

A entrada de dados do preditor *GMC* consiste de perfis *PSI_Blast* das sequências obtidas como parte do processo de busca sobre o banco não redundante de proteínas do *NCBI* (BENSON & WATERMAN, 1994). Além dos perfis *PSI_Blast* e dos perfis de frequência obtidos através da execução do programa com os parâmetros *default*, também foram executados testes com o perfis *PSI_Blast* gerados filtrando as regiões *coiled-coils* e de baixa complexidade das sequências presentes no banco do *NCBI*. Linhas adjacentes das matrizes dos perfis foram utilizadas como entrada, representando janelas de tamanho 13 nas sequências. O método aplicado sobre os bancos de dados na fase de treinamento e teste foi o *seven-fold cross validation*, sendo Q_3 a medida de desempenho escolhida. Ao invés do algoritmo *backpropagation* clássico, foi utilizada uma variação eficiente deste, o *RPROP* (RIEDMILLER & BRAUN, 1993), para treinamento e teste das três redes que compõem o preditor. As redes, com 30, 35 e 40 nós na camada escondida tiveram suas saídas combinadas através de cinco regras (TRESP, 2001): Votação, Produto, Média, Máximo e Mínimo como apresentado na Figura 4.11.

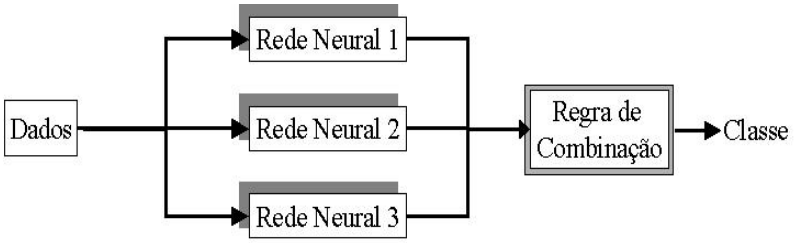

Figura 4.11: Arquitetura do preditor *GMC*

Os melhores resultados foram alcançados com os perfis *PSI_Blast* obtidos com os parâmetros *default* e combinando as saídas. Nessas condições, o desempenho médio Q_3 alcançado para o *RS126* foi de 74.13% e de 75.93% para o *CB396*. Os resultados apresentaram pouca variabilidade que pode ser demonstrada pelo baixo desvio padrão calculado para ambos os experimentos: menos de 1.5% para o *RS126* e próximo de 2% para o *CB396*. Ao compararmos com os resultados reportados no trabalho de Cuff e Barton (CUFF & BARTON, 2000), o índice alcançado pelo *GMC* com o *RS126* foi superado apenas pelo *CONSENSUS* que atingiu 74.8%. A diferença de 0.7% no entanto, não é significativa principalmente se levarmos em consideração que tal preditor é o resultado da combinação de quatro outros. O percentual obtido com o *CB396* por sua vez é o melhor resultado publicado para este banco de dados superando em três pontos percentuais o desempenho obtido com o *CONSENSUS*.

Pode-se observar a partir do estudo descrito brevemente nesta seção que, embora diversas abordagens tenham sido desenvolvidas, existem ainda muitas possibilidades a serem consideradas no problema da predição de estrutura secundária. As diretivas também são distintas, buscando alcançar os limites atuais usando metodologias que envolvam menos esforço computacional, ou superá-los uma vez que uma solução satisfatória para o problema ainda não foi desenvolvida (ROST, 2001). Dentre as possibilidades que vêm sendo exploradas podemos citar a extração de características dos dados de entrada através de técnicas computacionais ou biológicas (MELO, CAVALCANTI *et al.*, 2003), (WANG, MA *et al.*, 2001), a utilização de bancos de dados mais significativos, a aplicação de novas técnicas de redes neurais (POLLASTRI, PRZYBYLSKI *et al.*, 2002) ou mesmo o emprego de técnicas alternativas, como *Support Vector Machines* (HUA & SUN, 2001).

Em resumo, diferentes estratégias têm sido implementadas para melhorar o desempenho das redes na predição de estruturas secundárias de proteínas. Essas estratégias podem ser agrupadas em cinco grupos principais:

- Adicionar novos tipos de informações biológicas, por exemplo: informações evolucionárias, informações sobre a cadeia principal da molécula e dados sobre contatos entre aminoácidos adjacentes. Nesse caso não se utiliza apenas a homologia da sequência primária como fonte de informação para a *RNA*;
- Alteração da forma de como apresentar a informação para a *RNA*, por exemplo: estratégias que modificam o conjunto de treinamento;
- Utilizar filtros antes da predição e realizar pós-processamento;
- Alterar a arquitetura *feedfoward* padrão utilizada; e
- Treinamento balanceado.

Todas essas estratégias alteram, significativamente, o desempenho das *RNAs* na predição e podem ser combinadas e utilizadas conjuntamente. Existem diferentes linhas quando se discute como melhorar o desempenho das redes neurais. Alguns autores exploram o problema de melhorar a base de dados, outros autores exploram também o uso de informações evolutivas obtidas pelo alinhamento de sequências e outros exploram ainda que um bom projeto de rede neural seja um fator chave para o desempenho da predição.

A proposta deste trabalho é realizar algumas destas estratégias de melhoria, sendo algumas em conjunto, para alcançar uma maior acurácia de predição de estruturas secundárias de proteínas.

Capítulo V – Materiais e Métodos

Este capítulo apresenta os materiais e métodos utilizados nesta pesquisa. As arquiteturas da *RNA* que serão utilizadas neste trabalho serão detalhadas juntamente com os seus respectivos conjuntos de dados. São elas, a saber: uma Rede Neural Artificial Multicamadas com treinamento *backpropagation* e inicialização aleatória convencional, aqui denominada *cMLP*; outra com inicialização não-aleatória utilizando o critério pesos de Fisher, denominada *oMLP*. Posteriormente a este capítulo poder-se-á ter uma forma de avaliar a acurácia das *RNAs* desenvolvidas com os principais trabalhos publicados na comunidade científica e respectivos preditores disponíveis na *Web*.

5.1 Etapas do Projeto de Redes Neurais

Para o desenvolvimento de um modelo ou projeto baseado em redes neurais artificiais para predição da estrutura secundária de proteínas são necessárias diversas etapas:

a) **Coleta de dados de treinamento, validação e teste:** devem ser reunidos todos os dados pertinentes e potencialmente úteis à tarefa, isto é, deve-se reunir o conjunto de proteínas que será utilizado na classificação;
b) **Separação em conjuntos e pré-processamento dos dados:** os dados simbólicos (sequencias de aminoácidos) devem ser transformados em dados puramente numéricos (vetores binários de entrada), os quais são mais adequados para a utilização da rede;
c) **Projeto da estrutura e configuração da rede:** a escolha da configuração adequada da rede tem um impacto substancial no desempenho do sistema;
d) **Treinamento, teste e validação:** o treinamento é realizado em diferentes arquiteturas de rede e para estas arquiteturas se realiza a validação do treinamento e os testes para avaliar o melhor desempenho;
e) **Júri de decisão:** fase em que os resultados finais são gerados.

As fases acima descritas serão reorganizadas para melhor apresentação deste trabalho. Primeiramente serão apresentados os materiais, isto é, será abordada a

elaboração da base de dados, tanto para treinamento, como validação e teste. Após serão apresentados os métodos escolhidos para esta pesquisa: as duas arquiteturas de redes: *cMLP* e *oMLP*. Em resumo, a seguir serão apresentados todos os passos realizados para a criação das redes neurais artificiais que realizam a predição de estruturas secundárias de proteínas.

5.2 Base de Dados

Os dois primeiros passos do processo de desenvolvimento de redes neurais artificiais são: a coleta de dados relativos ao problema e a sua separação em conjunto de treinamento e conjunto de testes. Esta tarefa requer análise cuidadosa sobre o problema para minimizar ambiguidades e erros nos dados. Além disso, os dados coletados devem ser significativos e cobrir amplamente o domínio do problema; não devem cobrir apenas as operações normais ou rotineiras, mas também as exceções e as condições nos limites do domínio do problema.

Normalmente, os dados coletados são separados em duas categorias: dados de treinamento, que serão utilizados para o treinamento da rede e dados de teste, que serão utilizados para verificar seu desempenho sob condições reais de utilização. Além dessa divisão, pode-se usar também uma subdivisão do conjunto de treinamento, criando um conjunto de validação, utilizado para verificar a eficiência da rede quanto a sua capacidade de generalização durante o treinamento, e podendo ser empregado como critério de parada do treinamento.

Depois de determinados estes conjuntos, eles são, geralmente, colocados em ordem aleatória para prevenção de tendências associadas à ordem de apresentação dos dados. Além disso, pode ser necessário pré-processar estes dados, através de normalizações, escalonamentos e conversões de formato para torná-los mais apropriados à sua utilização na rede.

Nesta pesquisa procurou-se utilizar bases de dados já abordadas em trabalhos anteriores, tanto na fase de treinamento, quanto na fase de teste. Leva-se em consideração que os cuidados necessários em relação à escolha dos dados, como descrito acima, já foram tomados por estes trabalhos científicos clássicos. Isto porque a contribuição do trabalho não está na escolha da base de dados e sim na otimização da configuração da rede neural.

5.2.1 Proteínas utilizadas para Treinamento e Validação

Neste trabalho, o primeiro passo para a implementação do projeto foi a coleta e a seleção de um conjunto de sequências primárias de proteínas para utilizá-lo no treinamento da rede neural. Este conjunto de proteínas foi selecionado a partir do *PDB – Protein Data Bank* (http://www.rcsb.org/pdb) através de alinhamento múltiplo. Foram selecionadas ao todo 106 proteínas, destas 102 provenientes do trabalho de Qian & Sejnowski (1988) e outras 14 descritas nos testes do trabalho de Holley & Karplus (1991), com 10 destas fazendo interseção entre as duas bases e totalizando assim 106 proteínas diferentes que são responsáveis pelo treinamento das diferentes arquiteturas de *RNA*. As proteínas selecionadas seguem organizadas na Tabela 5.1.

Tabela 5.1: Proteínas utilizadas no Treinamento das Redes Neurais

				Classificação DSSP						
Proteína	Cadeias	Clas. SCOP	Clas. CATH	Total de Res.	Res. Alpha	Qtd. Hélices	% Alpha	Res. Beta	Qtd. Folhas	%Beta
1ABE	1	A/B	ALPHA BETA	306	145	15	47	67	14	22
1ACX	1	ALL BETA	MAINLY BETA	108	0	0	0	47	10	44
2APR	1	ALL BETA	MAINLY BETA	325	45	10	14	151	31	46
2AZAa	1 DE 2	ALL BETA	MAINLY BETA	129	21	4	16	46	11	36
2AZAb	2 DE 2	ALL BETA	MAINLY BETA	129	22	5	17	44	11	34
1AZU	1	ALL BETA	MAINLY BETA	128	14	2	11	35	10	27
1BP2	1	ALL ALPHA	MAINLY ALPHA	123	60	7	49	11	5	9
1CA2	1	ALL BETA	MAINLY BETA	259	42	10	16	77	18	30
1CC5	1	ALL ALPHA	MAINLY ALPHA	83	39	5	47	2	2	2
1CCR	1	ALL ALPHA	MAINLY ALPHA	111	47	6	42	2	2	2
5CPV	1	ALL ALPHA	MAINLY ALPHA	108	61	8	56	4	2	4
1CRN	1	SMALL PROTEINS	ALPHA BETA	46	22	3	48	4	2	9
1CTX	1	SMALL PROTEINS	MAINLY BETA	71	4	1	6	16	3	23
2CY3	1	ALL ALPHA	ALPHA BETA	118	30	4	25	8	6	7
1CYCa	1 DE 2	ALL ALPHA	MAINLY ALPHA	103	35	3	34	2	2	2
1CYCb	2 DE 2	ALL ALPHA	MAINLY ALPHA	103	35	3	34	2	2	2
1ECD	1	ALL ALPHA	MAINLY ALPHA	136	104	10	76	0	0	0
1EST	1	ALL BETA	MAINLY BETA	240	25	6	10	94	26	39
1FC2c	1 DE 2	ALL ALPHA	MAINLY ALPHA	58	21	2	36	0	0	0
1FC2d	2 DE 2	ALL BETA	MAINLY BETA	224	18	4	8	95	19	42
1FDHa	1 DE 4	ALL ALPHA	MAINLY ALPHA	141	99	8	70	0	0	0
1FDHb	2 DE 4	ALL ALPHA	MAINLY ALPHA	141	99	8	70	0	0	0
1FDHg	3 DE 4	ALL ALPHA	MAINLY ALPHA	146	108	10	74	0	0	0
1FDHh	4 DE 4	ALL ALPHA	MAINLY ALPHA	141	108	10	77	0	0	0
1DUR	1	A+B	ALPHA BETA	55	8	2	15	10	4	18

1FX1	1	A/B	ALPHA BETA	148	47	5	32	34	10	23
1GCN	1	PEPTIDES	-	29	14	2	48	0	0	0
4GCR	1	ALL BETA	MAINLY BETA	174	16	4	9	84	18	48
1GP1a	1 DE 2	A/B	ALPHA BETA	198	59	9	30	33	11	17
1GP1b	2 DE 2	A/B	ALPHA BETA	198	59	10	30	33	11	17
1HDSa	1 DE 4	ALL ALPHA	MAINLY ALPHA	141	81	9	57	1	1	1
1HDSb	2 DE 4	ALL ALPHA	MAINLY ALPHA	145	80	8	55	0	0	0
1HDSc	3 DE 4	ALL ALPHA	MAINLY ALPHA	141	96	11	68	0	0	0
1HDSd	4 DE 4	ALL ALPHA	MAINLY ALPHA	145	79	11	54	0	0	0
1HIP	1	SMALL PROTEINS	FEW SS	85	19	5	22	14	8	16
2HMQa	1 DE 4	ALL ALPHA	MAINLY ALPHA	113	79	6	70	0	0	0
2HMQb	2 DE 4	ALL ALPHA	MAINLY ALPHA	113	79	6	70	0	0	0
2HMQc	3 DE 4	ALL ALPHA	MAINLY ALPHA	113	79	6	70	0	0	0
2HMQd	4 DE 4	ALL ALPHA	MAINLY ALPHA	113	79	6	70	0	0	0
2IG2h	1 DE 2	ALL BETA	MAINLY BETA	455	16	5	4	104	25	23
2IG2l	2 DE 2	ALL BETA	MAINLY BETA	216	14	3	6	95	23	44
4INSa	1 DE 4	SMALL PROTEINS	-	21	12	3	57	2	2	10
4INSb	2 DE 4	SMALL PROTEINS	-	30	14	2	47	4	2	13
4INSc	3 DE 4	SMALL PROTEINS	-	21	12	3	57	2	2	10
4INSd	4 DE 4	SMALL PROTEINS	-	30	14	2	47	4	2	13
2LDXa	1 DE 4	A+B	ALPHA BETA	331	127	12	38	62	14	19
2LDXb	2 DE 4	A+B	ALPHA BETA	331	127	12	38	62	14	19
2LDXc	3 DE 4	A+B	ALPHA BETA	331	127	12	38	62	14	19
2LDXd	4 DE 4	A+B	ALPHA BETA	331	127	12	38	62	14	19
1LZ1	1	A+B	MAINLY ALPHA	130	51	7	39	16	9	12
2LZM	1	A+B	MAINLY ALPHA	164	109	10	66	15	4	9
1LZT	1	A+B	MAINLY ALPHA	129	55	7	43	14	9	11
1MBD	1	ALL ALPHA	MAINLY ALPHA	153	119	10	78	0	0	0
1MBS	1	ALL ALPHA	MAINLY ALPHA	153	111	8	73	0	0	0
2MLTa	1 DE 2	PEPTIDES	-	26	23	2	88	0	0	0
2MLTb	2 DE 2	PEPTIDES	-	26	24	1	92	0	0	0
1NXB	1	SMALL PROTEINS	MAINLY BETA	62	0	0	0	26	5	42
1P2P	1	ALL ALPHA	MAINLY ALPHA	124	54	6	44	9	5	7
1PFC	1	ALL BETA	MAINLY BETA	113	4	1	4	35	7	31
1PPD	1	A+B	ALPHA BETA	212	56	7	26	45	17	21
1PPT	1	PEPTIDES	-	36	18	1	50	0	0	0
1PYPa	1 DE 2	ALL BETA	ALPHA BETA	285	42	6	15	36	13	13
1PYPb	2 DE 2	ALL BETA	ALPHA BETA	285	42	6	15	36	13	13
1REIa	1 DE 2	ALL BETA	MAINLY BETA	107	3	1	3	52	11	49
1REIb	2 DE 2	ALL BETA	MAINLY BETA	107	3	1	3	54	11	50
1RHD	1	A/B	ALPHA BETA	293	87	12	30	39	17	13
3RN3	1	A+B	ALPHA BETA	124	26	4	21	44	10	35
2SN3	1	SMALL PROTEINS	ALPHA BETA	65	8	1	12	16	7	25
1TIMa	1 DE 2	A/B	ALPHA BETA	247	113	13	46	44	10	18
1TIMb	2 DE 2	A/B	ALPHA BETA	247	114	12	46	43	10	17

1TGSi	1 DE 2	SMALL PROTEINS	ALPHA BETA	56	9	1	16	11	4	20
1TGSz	2 DE 2	ALL BETA	MAINLY BETA	229	23	4	10	90	18	39
2ACT	1	A+B	ALPHA BETA	220	66	9	30	49	17	22
3ADK	1	A/B	ALPHA BETA	194	106	10	55	25	5	13
2ALP	1	ALL BETA	MAINLY BETA	198	14	3	7	109	22	55
4APE	1	ALL BETA	MAINLY BETA	330	31	7	9	154	27	47
3APP	1	ALL BETA	MAINLY BETA	323	45	11	14	155	32	48
1CYO	1	A+B	ALPHA BETA	93	29	5	31	20	6	22
2CAB	1	ALL BETA	ALPHA BETA	260	40	9	15	81	18	31
2CCYa	1 DE 2	ALL ALPHA	MAINLY ALPHA	128	95	6	74	2	2	2
2CCYb	2 DE 2	ALL ALPHA	MAINLY ALPHA	128	95	6	74	2	2	2
2CDV	1	ALL ALPHA	ALPHA BETA	107	30	5	28	12	6	11
2CYP	1	ALL ALPHA	MAINLY ALPHA	294	147	18	50	22	11	7
2DHBa	1 DE 2	ALL ALPHA	MAINLY ALPHA	141	86	9	61	0	0	0
2DHBb	2 DE 2	ALL ALPHA	MAINLY ALPHA	146	99	8	68	0	0	0
5FD1	1	A+B	ALPHA BETA	106	37	8	35	13	5	12
2GCHe	1 DE 3	-	-	13	0	0	0	0	0	0
2GCHf	2 DE 3	ALL BETA	MAINLY BETA	131	3	1	2	49	12	37
2GCHg	3 DE 3	ALL BETA	MAINLY BETA	97	20	4	21	35	7	36
2GN5	1	ALL BETA	MAINLY BETA	87	0	0	0	9	7	10
3GRS	1	A/B	ALPHA BETA	478	158	18	33	116	28	24
3ICB	1	ALL ALPHA	MAINLY ALPHA	75	43	4	57	2	2	3
2KAIa	1 DE 3	ALL BETA	MAINLY BETA	80	3	1	4	37	9	46
2KAIb	2 DE 3	ALL BETA	MAINLY BETA	152	18	3	12	43	9	28
2KAIi	3 DE 3	ALL BETA	MAINLY BETA	58	11	2	19	12	4	21
2LH1	1	ALL ALPHA	MAINLY ALPHA	153	119	9	78	0	0	0
2LHB	1	ALL ALPHA	MAINLY ALPHA	149	112	10	75	0	0	0
2MCPh	1 DE 2	ALL BETA	MAINLY BETA	222	9	3	4	113	22	51
2MCPl	2 DE 2	ALL BETA	MAINLY BETA	220	8	2	4	106	22	48
4MDHa	1 DE 2	A/B	ALPHA BETA	333	142	14	43	64	15	19
4MDHb	2 DE 2	A/B	ALPHA BETA	333	139	13	42	61	15	18
2PABa	1 DE 2	ALL BETA	MAINLY BETA	127	8	1	6	59	9	46
2PABb	2 DE 2	ALL BETA	MAINLY BETA	127	8	1	6	56	10	44
2RHE	1	ALL BETA	MAINLY BETA	114	3	1	3	52	12	46
2SBT	1	A/B	ALPHA BETA	275	59	7	21	38	10	14
2SGA	1	ALL BETA	MAINLY BETA	181	18	4	10	100	19	55
2SNS	1	ALL BETA	MAINLY BETA	149	29	4	19	32	12	21
2SODb	1 DE 4	ALL BETA	MAINLY BETA	151	0	0	0	55	11	36
2SODg	1 DE 4	ALL BETA	MAINLY BETA	151	4	1	3	58	12	38
2SODo	1 DE 4	ALL BETA	MAINLY BETA	151	3	1	2	64	15	42
2SODy	1 DE 4	ALL BETA	MAINLY BETA	151	4	1	3	59	12	39
3SSI	1	A+B	ALPHA BETA	113	17	3	15	35	7	31
2BUK	1	ALL BETA	MAINLY BETA	196	21	4	11	88	15	45
2TAAa	1 DE 3	A/B	ALPHA BETA	478	125	17	26	81	30	17
2TAAb	2 DE 3	A/B	ALPHA BETA	478	125	17	26	83	32	17

2TAAc	3 DE 3	A/B	ALPHA BETA	478	125	17	26	80	29	17
2TBVa	1 DE 3	ALL BETA	MAINLY BETA	387	7	2	2	114	26	29
2TBVb	2 DE 3	ALL BETA	MAINLY BETA	387	10	3	3	117	25	30
2TBVc	3 DE 3	ALL BETA	MAINLY BETA	387	7	2	2	116	25	30
3C2C	1	ALL ALPHA	MAINLY ALPHA	112	48	6	43	2	2	2
3CNA	1	ALL BETA	MAINLY BETA	237	0	0	0	96	17	41
4FXC	1	A+B	ALPHA BETA	98	15	3	15	32	7	33
3GPDg	1 DE 2	A+B	ALPHA BETA	334	92	10	28	73	19	22
3GPDr	2 DE 2	A+B	ALPHA BETA	334	92	10	28	73	19	22
3HHBa	1 DE 4	ALL ALPHA	MAINLY ALPHA	141	108	10	77	0	0	0
3HHBb	2 DE 4	ALL ALPHA	MAINLY ALPHA	146	118	11	81	0	0	0
3HHBc	3 DE 4	ALL ALPHA	MAINLY ALPHA	141	108	10	77	0	0	0
3HHBd	4 DE 4	ALL ALPHA	MAINLY ALPHA	146	118	10	81	0	0	0
3PCY	1	ALL BETA	MAINLY BETA	99	10	3	10	37	10	37
3PGK	1	A/B	ALPHA BETA	415	143	14	34	48	16	12
3PGMa	1 DE 2	A/B	ALPHA BETA	244	69	8	28	18	8	7
3PGMb	2 DE 2	A/B	ALPHA BETA	244	69	8	28	18	8	7
3RP2a	1 DE 2	ALL BETA	MAINLY BETA	224	18	3	8	89	21	40
3RP2b	2 DE 2	ALL BETA	MAINLY BETA	224	18	3	8	82	20	37
3SGBe	1 DE 2	ALL BETA	MAINLY BETA	185	12	2	6	98	19	53
3SGBi	2 DE 2	SMALL PROTEINS	ALPHA BETA	56	10	1	18	12	5	21
8TLNd	1 DE 2	-	-	2	0	0	0	0	0	0
8TLNe	2 DE 2	A/B	ALPHA BETA	316	131	12	41	54	17	17
451C	1	ALL ALPHA	MAINLY ALPHA	82	41	5	50	4	4	5
4CTSa	1 DE 2	ALL ALPHA	MAINLY ALPHA	437	229	23	52	23	10	5
4CTSb	2 DE 2	ALL ALPHA	MAINLY ALPHA	437	243	23	56	13	5	3
4DFRa	1 DE 2	A/B	ALPHA BETA	159	42	7	26	51	10	32
4DFRb	2 DE 2	A/B	ALPHA BETA	159	38	7	24	56	10	35
2FOX	1	A/B	ALPHA BETA	138	53	6	38	31	8	22
4SBVa	1 DE 3	ALL BETA	MAINLY BETA	260	30	6	12	73	12	28
4SBVb	2 DE 3	ALL BETA	MAINLY BETA	260	44	10	17	74	11	28
4SBVc	3 DE 3	ALL BETA	MAINLY BETA	260	38	8	15	74	12	28
5AT1a	1 DE 4	A/B	ALPHA BETA	310	118	14	38	49	14	16
5AT1b	2 DE 4	A/B	ALPHA BETA	153	21	4	14	46	10	30
5AT1c	3 DE 4	A/B	ALPHA BETA	310	123	15	40	49	14	16
5AT1d	4 DE 4	A/B	ALPHA BETA	153	25	5	16	52	10	34
5CPA	1	A/B	ALPHA BETA	307	117	11	38	52	10	17
5LDHa	1 DE 2	A/B	ALPHA BETA	333	130	12	39	34	13	10
5LDHb	2 DE 2	A/B	ALPHA BETA	333	130	12	39	34	13	10
5PTI	1	SMALL PROTEINS	FEW SS	58	12	2	21	15	3	26
5RXN	1	SMALL PROTEINS	MAINLY BETA	54	9	3	17	12	7	22
6ADHa	1 DE 2	A/B	ALPHA BETA	374	67	13	18	81	25	22
6ADHb	2 DE 2	A/B	ALPHA BETA	374	66	12	18	75	24	20
8APIa	1 DE 2	A/B	ALPHA BETA	347	103	10	30	117	12	34
8APIb	2 DE 2	-	-	36	0	0	0	18	6	50

8CATa	1 DE 2	MULTIDOMAIN	-	506	162	22	32	84	19	17
8CATb	2 DE 2	MULTIDOMAIN	-	506	158	21	31	84	19	17
1GPDg	1 de 2	A/B	ALPHA BETA	333	87	7	26	75	23	23
1GPDr	2 de 2	A/B	ALPHA BETA	333	88	8	26	79	20	24
8ADH	1	A/B	ALPHA BETA	374	107	17	29	91	23	24
1LH1	1	ALL ALPHA	MAINLY ALPHA	153	119	9	78	0	0	0
1OVOa	1 DE 4	SMALL PROTEINS	ALPHA BETA	56	10	1	18	12	4	21
1OVOb	2 DE 4	SMALL PROTEINS	ALPHA BETA	56	10	1	18	12	4	21
1OVOc	3 DE 4	SMALL PROTEINS	ALPHA BETA	56	9	1	16	9	3	16
1OVOd	4 DE 4	SMALL PROTEINS	ALPHA BETA	56	9	1	16	9	3	16

Deve-se perceber que a maioria das proteínas apresenta mais do que uma cadeia e neste trabalho, foram utilizadas todas as cadeias de todas as proteínas, totalizando 170 sequencias de aminoácidos, 32.049 resíduos, sendo classificados pelo software *DSSP* (http://www.sander.ebi.ac.uk/DSSP): 9.657 resíduos alpha e 6.798 resíduos beta.

Depois de construído os conjuntos de sequências primárias, independentemente se são de treinamento, validação ou teste, é preciso obter os arquivos *PDB* referentes a essas proteínas. Esses arquivos contêm informações sobre a estrutura terciária da proteína, coordenadas atômicas, estruturas secundária e primária, entre outras. Para se conseguir a estrutura secundária dos três conjuntos, utilizou-se o *software DSSP* que classifica as estruturas secundárias em características geométricas e à exposição dos resíduos ao solvente.

O *DSSP* é o programa padrão para obtenção da estrutura secundária de sequências. Ele, através da entrada dos arquivos *.pdb*, analisa a geometria e os parceiros das pontes de hidrogênio da estrutura principal de cada resíduo em uma estrutura proteica conhecida, produzindo uma saída tabular que inclui numeração de resíduos, sequências, pontes de hidrogênio e detalhes geométricos, através de arquivos *.dssp*. Estes arquivos serão utilizados como vetores de saída das redes neurais projetadas.

Esse programa classifica cada resíduo em oito classes: H = α-hélices; G = *3*-hélices ou *3/10*-hélices; I = *5*-hélices *pi*; B = folhas-β isoladas; E = folhas-β; T = retorno ou volta; S = curva; e "." = indefinido. Essas classes são, tipicamente, integradas em três classes padronizadas em: H, G e I em α-hélices; B e E em folhas-β; e S, T e "." em *coils*.

Outros dois bancos de dados de classificação de estruturas de proteínas foram verificados: *SCOP* e *CATH*, nas terceira e quarta colunas, respectivamente.

O *SCOP – Structural Classification Of Proteins* (http://scop.mrc-lmb.cam.ac.uk/scop/) é um banco de dados mantido pelo *MRC Laboratory of Molecular Biology* em Cambridge. Neste banco, as proteínas conhecidas são geralmente agrupadas por suas características da estrutura secundária em: todas alfa, todas beta, alças, pequenas proteínas com íons estruturais de metal e diversos tipos de estruturas alfa-beta. Esses tipos principais são denominados classes dentro do *SCOP*. Além do nível classe, as proteínas são também divididas em: *Fold*, *Family* e *Superfamily*, no *SCOP*. Na Figura 5.1 segue a interface web do programa.

A interface *CATH* (http://www.cathdb.info/) é similar à *SCOP* em conceito, mas divide o *PDB* de modo um pouco diferente. Em *CATH*, as proteínas são classificadas em nível de (C)lasse, (A)rquitetura, (T)opologia e superfamília (H)omóloga. A interface do software pode ser visualizada na Figura 5.2.

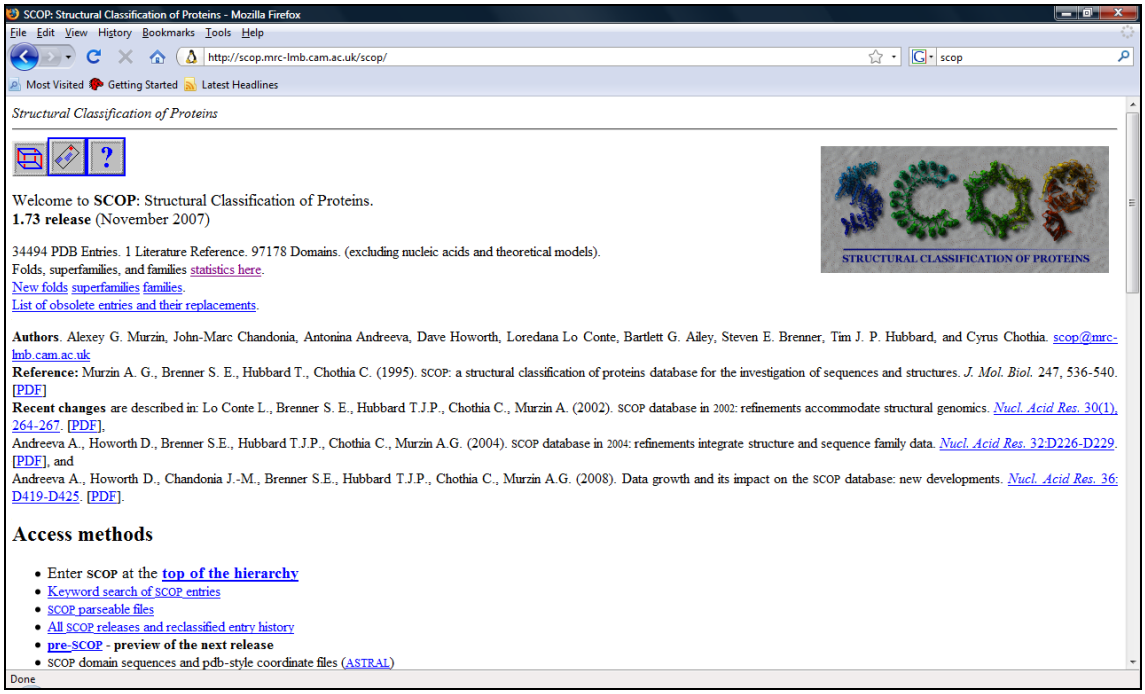

Figura 5.1: Interface *Web* do Banco de Dados *SCOP*

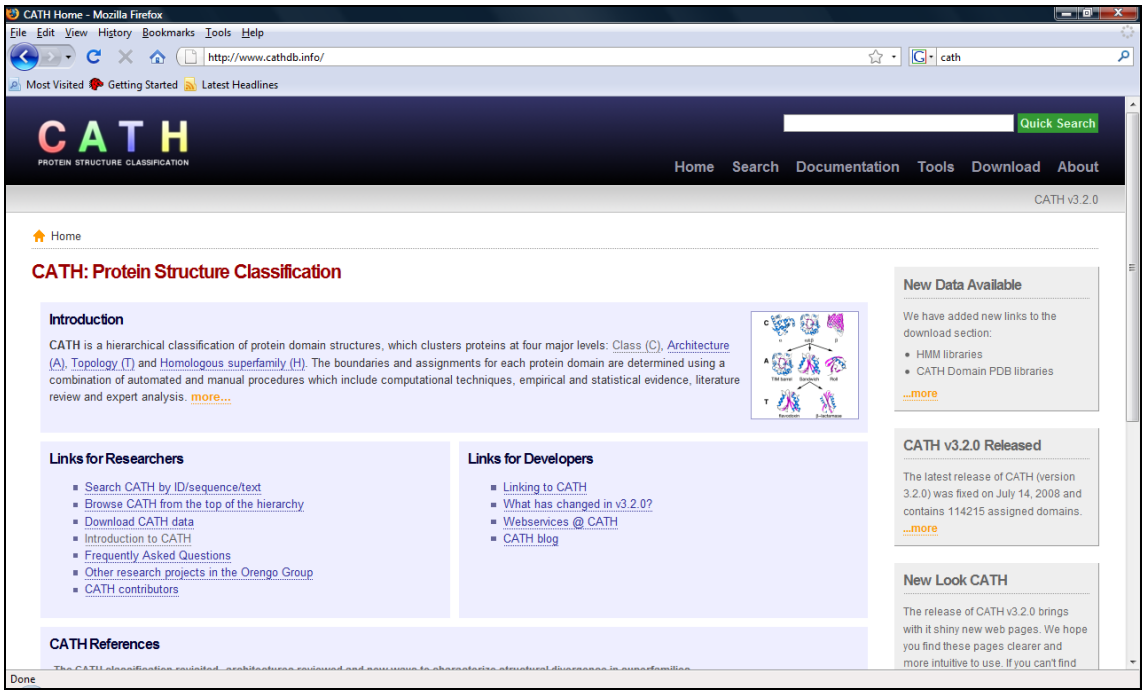

Figura 5.2: Interface *Web* do Banco de Dados *CATH*

Estas proteínas foram divididas em quatro subconjuntos para o treinamento das *RNA*'s:

a) ***Todas***, o qual possui todas as 106 proteínas coletadas;

b) ***Hélice***, que contém proteínas cujo número de resíduos em estrutura α-hélices é maior que a soma de resíduos em estrutura folhas-β e resíduos em estruturas *coil*. Farão parte deste subconjunto as proteínas classificadas como estruturas *All Alpha* no *SCOP* e *Mainly Alpha* no *CATH*;

c) ***Folha***, que contém proteínas cujo número de resíduos em estrutura folhas- β é maior que a soma de resíduos em estrutura α-hélices e resíduos em estruturas *coil*. Farão parte deste subconjunto as proteínas classificadas como estruturas *All Beta* no *SCOP* e no *Mainly Beta* no *CATH*; e

d) ***Hélice-Folha***, ou seja, proteínas cuja classificação seja *Alpha and Beta Proteins* no *SCOP* e *Alpha Beta* no *CATH*.

Um segundo conjunto de proteínas foi selecionado para validação da base de dados. O conjunto foi extraído do trabalho de Holley e Karplus (1991) e consiste das 14 proteínas utilizadas na fase de testes do trabalho. Essas proteínas seguem na Tabela 5.2.

Tabela 5.2: Proteínas utilizadas na Validação das Redes Neurais

				Classificação DSSP						
Proteína	**Cadeias**	**Clas. SCOP**	**Clas. CATH**	**Total de Res.**	**Res. Alpha**	**Qtd. Hélices**	**% Alpha**	**Res. Beta**	**Qtd. Folhas**	**%Beta**
1GPDg	1 de 2	A/B	ALPHA BETA	333	87	7	26	75	23	23
1GPDr	2 de 2	A/B	ALPHA BETA	333	88	8	26	79	20	24
8ADH	1	A/B	ALPHA BETA	374	107	17	29	91	23	24
3GRS	1	A/B	ALPHA BETA	478	158	18	33	116	28	24
2SODb	1 DE 4	ALL BETA	MAINLY BETA	151	0	0	0	55	11	36
2SODg	2 DE 4	ALL BETA	MAINLY BETA	151	4	1	3	58	12	38
2SODo	3 DE 4	ALL BETA	MAINLY BETA	151	3	1	2	64	15	42
2SODy	4 DE 4	ALL BETA	MAINLY BETA	151	4	1	3	59	12	39
1LH1	1	ALL ALPHA	MAINLY ALPHA	153	119	9	78	0	0	0
1CRN	1	SMALL PROTEINS	ALPHA BETA	46	22	3	48	4	2	9
1OVOa	1 DE 4	SMALL PROTEINS	ALPHA BETA	56	10	1	18	12	4	21
1OVOb	2 DE 4	SMALL PROTEINS	ALPHA BETA	56	10	1	18	12	4	21
1OVOc	3 DE 4	SMALL PROTEINS	ALPHA BETA	56	9	1	16	9	3	16
1OVOd	4 DE 4	SMALL PROTEINS	ALPHA BETA	56	9	1	16	9	3	16
3SSI	1	A+B	ALPHA BETA	113	17	3	15	35	7	31
1CTX	1	SMALL PROTEINS	MAINLY BETA	71	4	1	6	16	3	23
2MLTa	1 DE 2	PEPTIDES	-	26	23	2	88	0	0	0
2MLTb	2 DE 2	PEPTIDES	-	26	24	1	92	0	0	0
1NXB	1	SMALL PROTEINS	MAINLY BETA	62	0	0	0	26	5	42
3ADK	1	A/B	ALPHA BETA	194	106	10	55	25	5	13
1RHD	1	A/B	ALPHA BETA	293	87	12	30	39	17	13
2PABa	1 DE 2	ALL BETA	MAINLY BETA	127	8	1	6	59	9	46
2PABb	2 DE 2	ALL BETA	MAINLY BETA	127	8	1	6	56	10	44

5.2.2 Proteínas utilizadas para Teste

O conjunto de proteínas utilizado para testes foi obtido através do *CASP - Critical Assessment of Structure Prediction* (CASP, 2008). A Figura 5.3 apresenta a interface web do banco de dados *CASP*.

As 15 proteínas da Tabela 5.3 são usadas, como padrão, pelo *CASP* na avaliação dos métodos de predição de estrutura secundária descritos na literatura. Este mesmo conjunto de proteínas é encontrado nos trabalhos de Ferreira (2004) e Scott, Chahine & Ruggiero (2007) o que facilita a comparação entre os preditores.

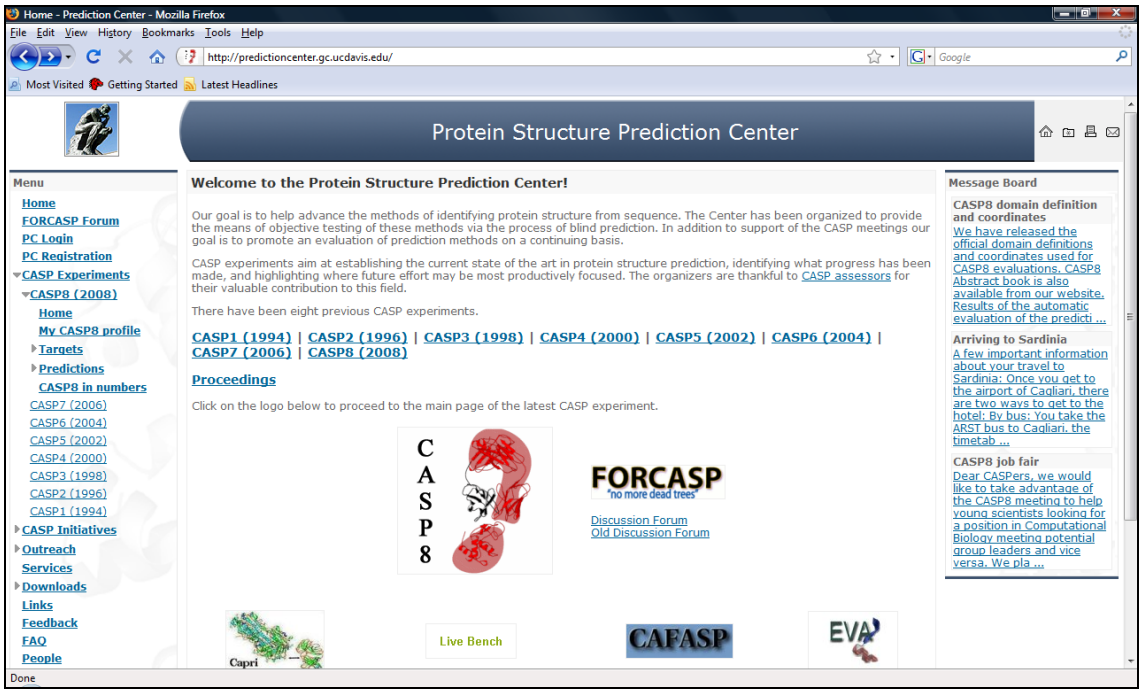

Figura 5.3: Interface *Web* do *CASP*

Tabela 5.3: Proteínas utilizadas no Teste das Redes Neurais

				Classificação DSSP						
Proteína	**Cadeias**	**Clas. SCOP**	**Clas. CATH**	**Total de Res.**	**Res. Alpha**	**Qtd. Hélices**	**% Alpha**	**Res. Beta**	**Qtd. Folhas**	**%Beta**
1QLQ	1	SMALL PROTEINS	FEW SS	58	12	2	21	15	3	26
1EIG	1	A+B	MAINLY BETA	73	13	1	18	14	3	19
1C56	1	SMALL PROTEINS	ALPHA BETA	40	11	1	28	9	3	23
1DAQ	1	ALL ALPHA	MAINLY ALPHA	71	24	3	34	0	0	0
1EHD	1	SMALL PROTEINS	-	89	20	6	22	20	10	22
1E5B	1	ALL BETA	MAINLY BETA	87	0	0	0	32	12	37
1EJG	1	SMALL PROTEINS	ALPHA BETA	46	19	2	41	4	2	9
1ES1	1	A+B	ALPHA BETA	82	28	6	34	18	7	22
1DT4	1	A+B	ALPHA BETA	73	31	4	42	20	3	27
1EDS	1	PEPTIDES	-	31	13	2	42	0	0	0
1G6X	1	SMALL PROTEINS	FEW SS	58	12	2	21	15	3	26
1DOI	1	A+B	ALPHA BETA	128	38	7	30	32	7	25
1FD8	1	A+B	ALPHA BETA	73	22	2	30	24	4	33
1FE5	1	ALL ALPHA	MAINLY ALPHA	118	53	5	45	10	4	8
1EHJ	1	ALL ALPHA	ALPHA BETA	68	22	4	32	7	3	10

Para verificar se as proteínas não possuíam um alto grau de identidade na sequência, foi realizado alinhamento múltiplo através do software *Clustal-X* (http://genome.jouy.inra.fr/doc/clustal/clustalx.html). Este software baseia-se no conceito de alinhamento progressivo, o qual determina os alinhamentos para cada par de

sequências e constrói uma matriz de distâncias que reflete estes alinhamentos. O alinhamento múltiplo é classificado como eficiente quando sua porcentagem de homologia for baixa (30% de homologia caracterizada bem alinhada).

5.2.3 Interface de Pré-Processamento

Com a definição do problema e a coleta dos dados concluída, segue-se à codificação dos dados de entrada das *RNA*s. Para isso, desenvolvemos um software chamado *CONFIGPRED* (Configurador dos dados para o Preditor de Estrutura Secundária de Proteínas através de Redes Neurais), desenvolvido em *Linguagem C*. Esse software desenvolve várias funções de pré-processamento, tais como:

- Conversão de vários arquivos *PDB* para arquivos *DSSP*.
- Obtenção das três classes *(α*-hélices, folhas-*β* e *coils*) que serão utilizadas;
- Divisão da base de treinamento em 4 subconjuntos de acordo com a sua porcentagem de estruturas: todas, hélice, folha e hélices/folhas;
- Criação de arquivos que descrevem as matrizes de treinamento e as janelas de aminoácidos, ou seja, a codificação da base de treinamento em matrizes separadas por janelas;
- Geração de arquivos estatísticos sobre as proteínas, ou seja, quantos resíduos de aminoácidos compõem a base de treinamento e qual tipo de estrutura é composta essa base;
- Geração de matrizes de entrada e saída para a utilização nas *RNA*s.

5.3 Preditores utilizados para comparação

Entre os métodos utilizados neste trabalho estão os preditores amplamente difundidos na *Web*, resultados de trabalhos acadêmicos bem sucedidos na literatura de Biologia Computacional. A ideia central é desenvolver um preditor e comparar seus resultados com os resultados obtidos por estes preditores. Seguem alguns preditores que foram utilizados para comparação.

5.3.1 PredictProtein

O *PredictProtein* (http://predictprotein.org), antigo *PHD – Predict secondary HeiDelberg* é resultado do trabalho de Rost, Yachdav & Liu (2004). Ele combina resultados de um número variável de redes neurais, onde cada uma prevê a estrutura

secundária de um resíduo baseado no contexto sequencial local e nas características sequenciais globais (comprimento da proteína, frequências de aminoácidos, entre outras). A predição final é uma média aritmética da saída de cada uma dessas redes neurais. Tais esquemas de combinações são conhecidos como decisão de júri. O *PredictProtein* é considerado padrão para a predição da estrutura secundária. A Figura 5.4 apresenta a interface *web* deste *software*.

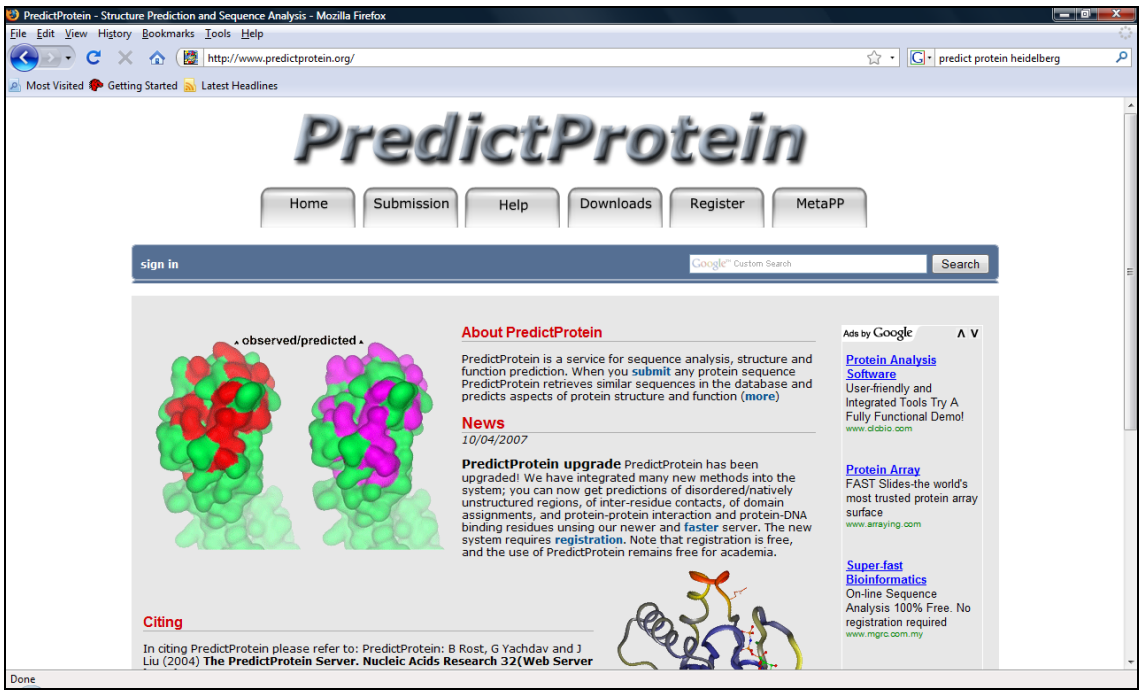

Figura 5.4: Interface *Web* do Programa *PredictProtein*

5.3.2 PSIPRED

O *PSIPRED* (http://bioinf.cs.ucl.ac.uk/psipred/) é resultado do trabalho de Jones (1999) e combina previsões de redes neurais com um alinhamento sequencial múltiplo alcançado por uma busca pelo banco de dados *PSI-BLAST*. O *PSIPRED* foi considerado pela *CASP* (CASP, 2008) um dos que melhor executou a predição da estrutura secundária. A Figura 5.5 apresenta a interface *web* do programa *PSIPRED*.

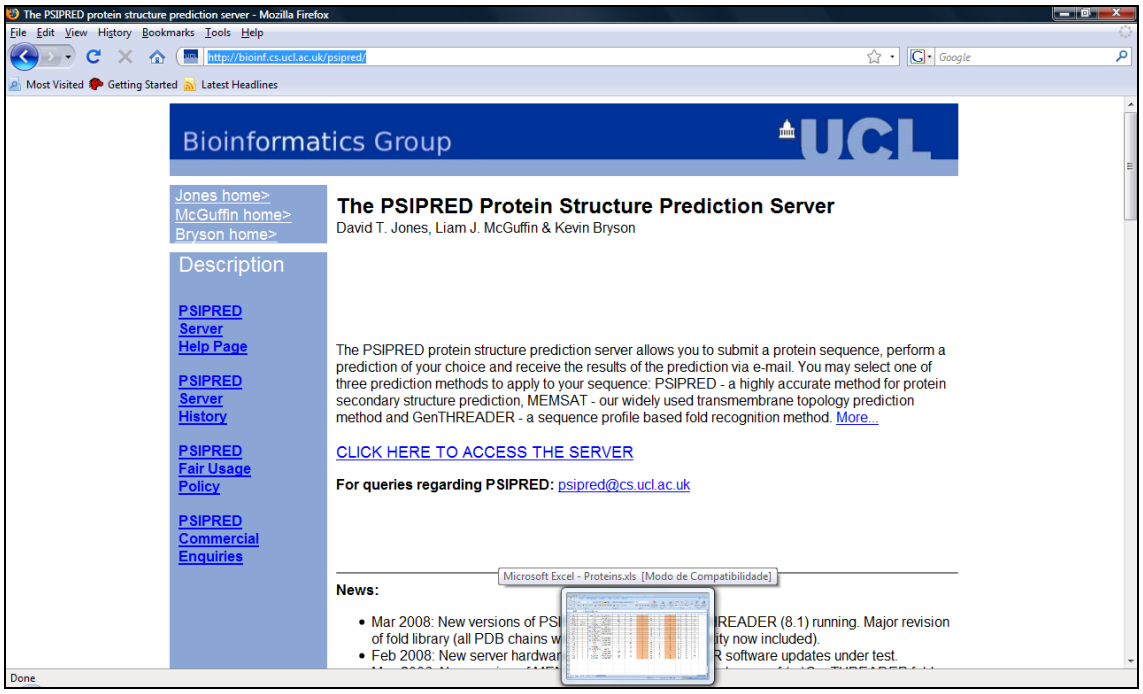

Figura 5.5: Interface *Web* do Programa *PSIPRED*

5.3.3 JPred

As predições de estrutura secundária do *JPred* (CUFF; CLAMP; SIDDIQUI; FINLAY & BARTON, 1998) partem do consenso de vários outros métodos complementares, completados pelas informações do alinhamento sequencial múltiplo. O servidor do *JPred* (http://www.compbio.dundee.ac.uk/~www-jpred) retorna o resultado que pode, por sua vez, ser exibido, editado e gravado para ser usado por outros programas com o editor de alinhamento *Jalview*. A Figura 5.6 apresenta a interface web do programa *JPred*.

5.3.4 PREDATOR

O *PREDATOR* (http://wwwTools.GroupLeftEMBL/argos/predator/down_predator.html) combina informações de alinhamento sequencial múltiplo com as características de ponte de hidrogênio dos aminoácidos para prever a estrutura secundária. Este software é resultado do trabalho de Frishman & Argos (1997). A Figura 5.7 apresenta a interface web do programa *Predator*.

Figura 5.6: Interface *Web* do Programa *JPred*

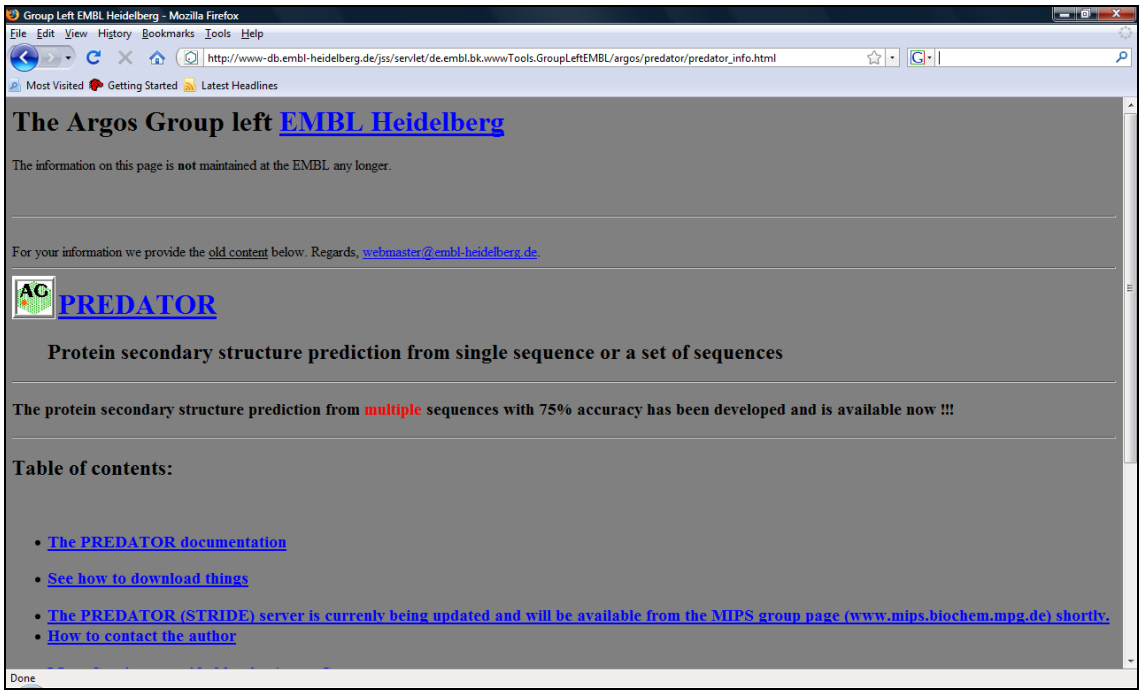

Figura 5.7: Interface *Web* do Programa *Predator*

5.3.5 PSA

O *PSA – Protein Sequence Analysis* é uma abordagem baseada nos modelos ocultos de Markov para a predição de estrutura secundária, que possui um notável resultado gráfico, rico em detalhes e representa as probabilidades previstas dos estados

de hélice, folha e *coil* para cada posição na sequência proteica. É resultado do trabalho de Stultz, Nambudripad, Lathrop & White (1997). A Figura 5.8 apresenta a interface web do programa *PSA*.

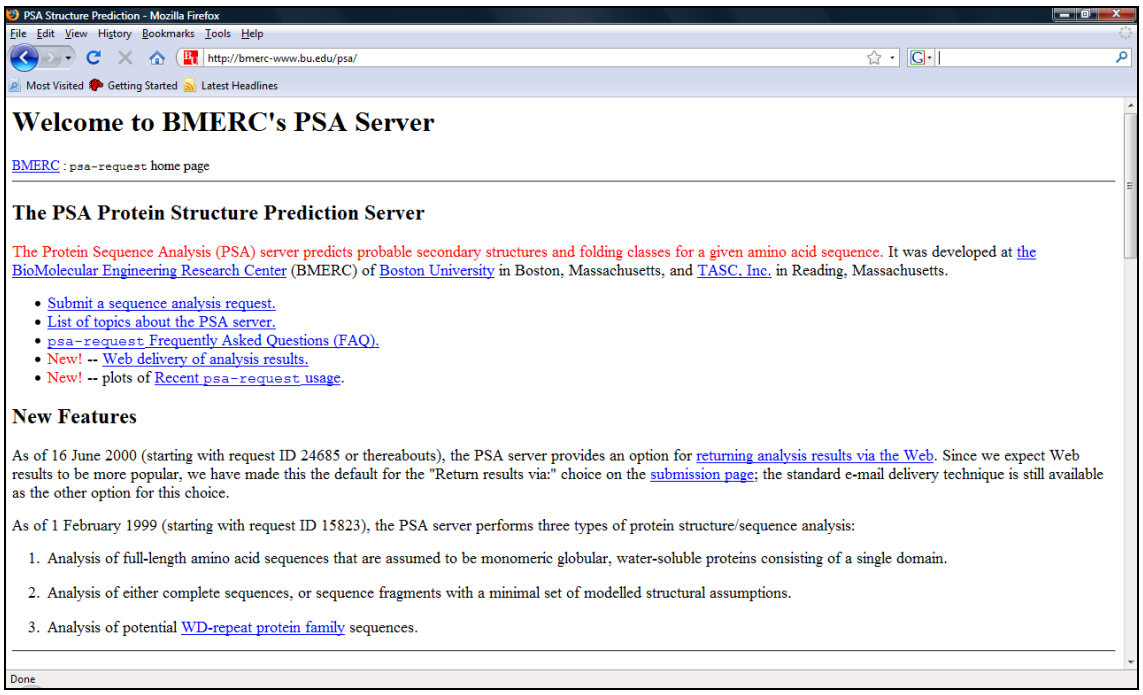

Figura 5.8: Interface *Web* do Programa *PSA*

5.3.6 PREDCASA

Além dos preditores anteriores, todos disponíveis na *Web*, ainda serão levados em consideração os resultados obtidos pelo preditor *PREDCASA* resultado dos trabalhos de Ferreira (2004) e de Scott, Chahine & Ruggiero (2007).

5.4 Arquitetura da Rede Neural *MLP* Convencional (Rede Neural *cMLP*)

O objetivo usual dos métodos de predição de estruturas secundárias é associar a cada resíduo um padrão estrutural, dependente do conjunto de resíduos que lhe são adjacentes, ou seja, de uma janela da sequência. A principal ideia é o fato de que segmentos de resíduos consecutivos possuem uma preferência para certos estados de estrutura secundária. Dessa forma, o problema de predição de estrutura torna-se um problema clássico de classificações de padrões tratável por algoritmos de reconhecimento de padrões, tais como redes neurais *MLP*.

Foram projetadas e implementadas duas arquiteturas diferentes: uma contendo apenas uma rede neural com três camadas intermediárias, pelo motivo de ter alcançado o melhor resultado no trabalho de Qian & Sejnowski (1988); e outra contendo duas redes neurais cada uma com três camadas intermediárias, pelo mesmo motivo. Sendo que, nesse caso, a saída da primeira rede alimenta a entrada da segunda rede neural. Para o caso da arquitetura com duas redes neurais, a informação de saída da primeira rede neural é adicionada na janela de dados de entrada.

Para cada arquitetura foram implementadas 9 redes *MLP* com a camada de entrada variando. Todas as redes foram projetadas de maneira a predizer a estrutura secundária do aminoácido central do janelamento. Foram testadas redes com os seguintes tamanhos de janela de entrada: 7, 9, 11, 13, 15, 17, 19, 21 e 23. Vale ressaltar que no trabalho de Qian & Sejnowski (1988) os melhores resultados foram alcançados com tamanho de janela 13. Pelos resultados obtidos pode-se perceber que tamanhos diferenciados de janelas de entrada estão diretamente relacionados com o desempenho da rede na predição da estrutura.

5.4.1 Implementação da Arquitetura com uma *RNA*

O passo seguinte é a definição da estrutura e configuração da rede, que pode ser dividido em três etapas: seleção da arquitetura da rede apropriada à aplicação; determinação da topologia da rede a ser utilizada, isto é, o número de camadas, o número de unidades em cada camada, entre outros; e determinação de parâmetros do algoritmo de treinamento e funções de ativação. Este passo tem um grande impacto no desempenho do sistema resultante. Existem metodologias e principalmente heurísticas na condução destas tarefas. Normalmente estas escolhas são feitas de forma empírica.

Configuração da camada de entrada (janelas) e da camada de saída da *RNA*

Um dos objetivos deste trabalho constitui-se no uso de janelas de aminoácidos distintas. Foram implementadas 9 (nove) *RNA's*, onde cada uma foi projetada com janela diferente. A escolha do tamanho diferenciado de janelas de entradas tem uma contribuição bem satisfatória no trabalho. Esta contribuição é atribuída ao fato que as proteínas possuem estruturas secundárias iguais, contudo de tamanhos diferentes.

Assim, as 9 (nove) janelas de entradas diferentes foram desenvolvidas com o objetivo de predizer a estrutura secundária do resíduo presente na posição central das janelas. O *software CONFIGPRED* cria arquivos de janelas distintas (7, 9, 11, 13, 15,

17, 19, 21 e 23), ou seja, para cada sequência de aminoácidos, o *software* cria 9 (nove) matrizes de entrada, sendo cada uma com formato diferente de janela. Os aminoácidos da sequência primária de cada proteína são codificados em dados numéricos binários, como realizado pela primeira vez no trabalho de Qian & Sejnowski (1988). Para cada resíduo de aminoácido foram atribuídas 22 (vinte e duas) unidades, sendo 20 (vinte) reservadas para os aminoácidos, uma para o heteroátomo (*X*) e a última reservada para indicar o fim da janela ou o aminoácido não identificado pelo *DSSP*. Uma dessas 22 (vinte e duas) unidades é sinalizada com o valor 1, identificando um determinado resíduo e as outras unidades são atribuídos o valor 0, como exemplificado na Tabela 5.4.

Tabela 5.4: A representação da codificação dos aminoácidos

1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**Alanina**
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**Arginina**
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	**Fim da seq.**

Na tabela 5.4, tem-se um exemplo de como os aminoácidos são codificados, formando assim as matrizes de entrada preenchidas por 0 (zero) e 1 (um). Cada matriz de entrada possui um tamanho fundamentado na codificação dos aminoácidos em 22 unidades multiplicada pelo tamanho da janela correspondente a cada *RNA*, como apresentado na Tabela 5.5.

Tabela 5.5: Número de neurônios na camada de entrada das *RNA's*

Unidades	Janelas	N° de neurônios na camada de entrada
22	7	22 x 7 = 154
22	9	22 x 9 = 198
22	11	22 x 11 = 242
22	13	22 x 13 = 286
22	15	22 x 15 = 330
22	17	22 x 17 = 374
22	19	22 x 19 = 418
22	21	22 x 21 = 462
22	23	22 x 23 = 506

Para realizar os treinamentos e testes da rede é preciso fornecer a saída para cada padrão (janela), conseguido através do software *DSSP*. A saída da rede é a estrutura referente ao resíduo central. Para codificar os dados simbólicos (estrutura α-hélices, folhas-β e *coils*) em dados numéricos, foram definidos para cada saída da rede três unidades, como apresentado na Tabela 5.6. A partir da Tabela 5.6, ressalta-se a

codificação dos três padrões, sendo que os resíduos com formação de α-hélices recebem o valor (1 0 0), folhas-β (0 0 1) e *coils* (0 1 0).

Tabela 5.6: Codificação binária das estruturas secundárias

Estrutura	Codificação
α-Hélices	1 0 0
Folhas-β	0 0 1
Coils	0 1 0

Configuração da Camada Intermediária, Funções de Ativação e Implementação da *RNA*

A rede neural *cMLP* (MORAIS, 2009) apresentada neste trabalho foi implementada, treinada e testada utilizando a Linguagem de Programação *C*, através do software *Bloodshed Dev-C++ 4.0*, em ambiente *MS-Windows*. A preferência pelo desenvolvimento completo da Rede Neural, preterindo softwares comerciais tais como *MATLAB*, objetiva o melhor entendimento da configuração da rede para posterior otimização de seus parâmetros.

Também, foi utilizada para validação do software construído em Linguagem *C*, a ferramenta *JavaNNS – Java Neural Network Simulator* (ZELL, 2005) em função dos recursos apresentados pela ferramenta especificação de redes neurais e amplo suporte à rede *backpropagation*. Esta ferramenta possui grandes facilidades no que tange a simulação, visualização e também implementação de redes neurais artificiais. O processo de automatização do processo de treinamento também é uma característica positiva.

As *RNAs* implementadas foram do tipo *MLP* (*MultiLayer Perceptron*) com o treinamento realizado pelo algoritmo *backpropagation*. Na elaboração de um projeto de *RNA*, a tarefa consiste em determinar o número de camadas intermediárias, bem como o número de neurônios de processamento da cada camada intermediária, embora não existam regras para determinar o número de camadas e de neurônios. Sendo assim, as redes foram projetadas com três camadas intermediárias, devido a resultados alcançados por trabalhos anteriores e uma camada de saída, de forma que a camada de entrada utiliza a função de ativação linear e as demais, função de ativação sigmoide.

Na Figura 5.9 tem-se um aminoácido codificado sendo processado por uma rede com janela de 17 resíduos. A primeira camada tem 374 neurônios, ou seja, 22 unidades vezes 17, em que o processo de cálculo dos pesos nessa camada é realizado pela função

linear. Os dados, após serem calculados, são transferidos à camada seguinte. A mesma recalcula os pesos e passa para a camada de saída por meio da função sigmoide.

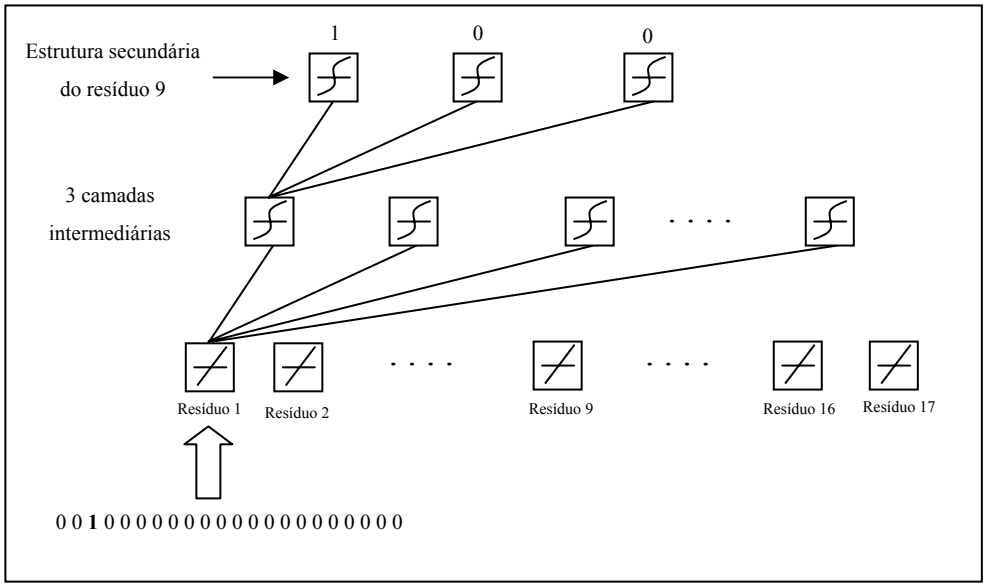

Figura 5.9: Arquitetura da *RNA* utilizada neste trabalho

Treinamento e Critério de Parada

Nesse processo, em princípio, os elementos de processamento das *RNAs* são inicializados aleatoriamente com diferentes valores de *thresholds* internos e com conexões de pequenos pesos, variando entre $-1 \leq w \leq -1$, em que w denota o peso de uma conexão qualquer. Em seguida, os dados existentes nas entradas e saídas do conjunto de treinamento são apresentadas às *RNAs*, repetidamente e, a cada ciclo de treinamento, os pesos são ajustados a fim de avaliar a generalização das redes. São dois os critérios de parada:

1. **Parada do treinamento no momento em que a rede atinge o objetivo:** o treinamento é interrompido quando o erro médio quadrático do treinamento for menor que 0,01 (1% de erro);
2. **Parada do treinamento depois de determinado ciclo:** o treinamento é interrompido depois de determinado número de ciclos, a fim de obter uma estimativa do erro da rede sobre o conjunto de validação.

Existem vários métodos para a determinação do momento em que o treinamento de uma rede neural deve ser encerrado. A determinação destes critérios é fundamental para um bom treinamento e, consequentemente uma boa generalização. Os critérios de parada heurísticos utilizados no projeto, em ordem de prioridade, são:

a) Treinar a rede *X* ciclos, com o conjunto de treinamento;
b) Simular a rede com a entrada do conjunto de validação;
c) Calcular o resultado contendo o módulo da diferença entre o resultado da simulação da validação e a saída real da validação;
d) Considerar como um erro de padrão da rede, quando cada linha da matriz de resposta tiver a soma superior a 0,5;
e) Somar o número de linhas do conjunto de validação que tiverem erro de padrão;
f) Parar o treinamento ou treinar *X* ciclos novamente, se no passo atual do conjunto validação contiver um número de linhas com erro ou esse erro for maior do que o número calculado.

5.4.2 Implementação da Arquitetura com duas *RNA's*

Com o objetivo de melhorar os resultados dos testes realizados com a arquitetura composta por uma *RNA*, foi implementada uma nova arquitetura utilizando duas redes, sendo que a saída da primeira rede é a entrada da segunda rede, conforme a Figura 5.10.

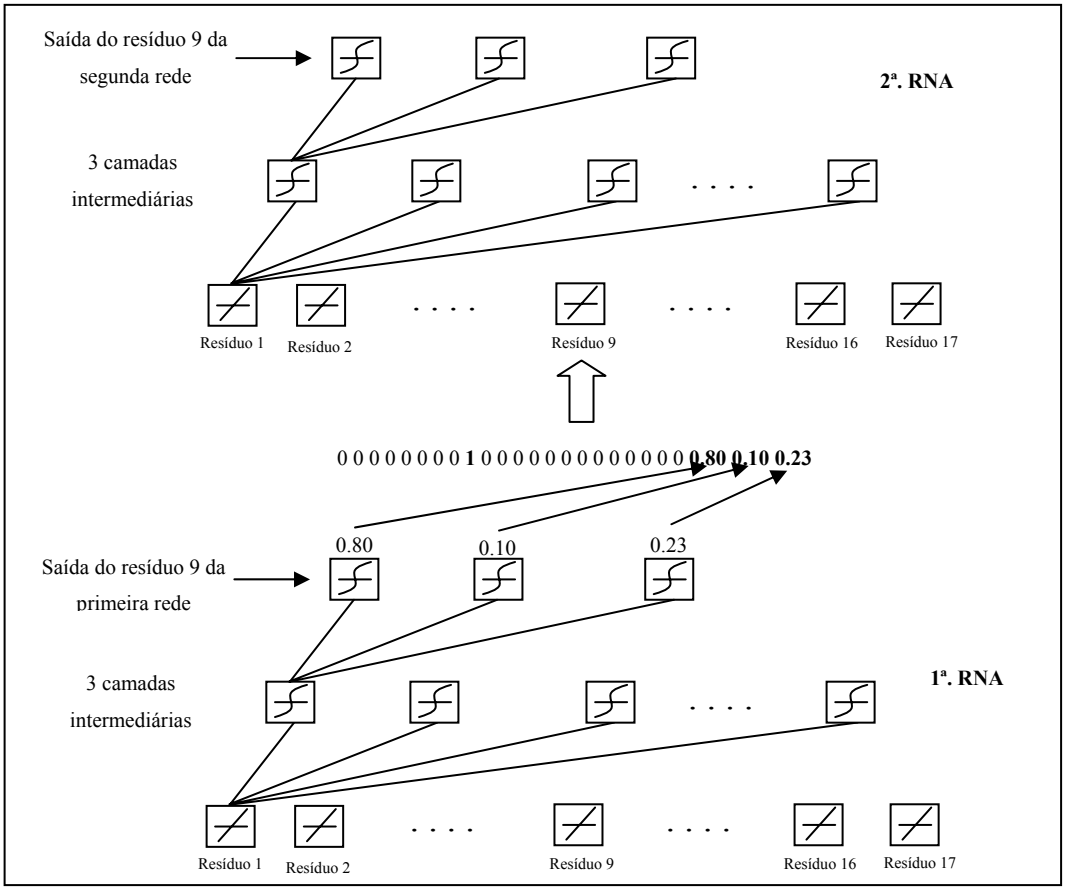

Figura 5.10: Arquitetura de duas *RNAs* utilizadas neste trabalho

É necessária a tarefa de conversão dos dados de entrada par a segunda rede. Essa tarefa é desenvolvida na inclusão de três colunas, cujos valores são o resultado da predição da estrutura secundária codificado na linha da matriz. Dessa forma, a nova matriz construída possuirá tanto o resíduo como também a sua estrutura secundária identificada pela rede (por exemplo, 0.80, 0.10, 0.23). Na inclusão dessas informações, a matriz original, de 22 (vinte e duas) unidades, passa a ter 25 (vinte e cinco).

Na Tabela 5.7, em que 18 arquiteturas de *RNAs* são formadas por suas janelas, sua estrutura padrão é composta por 24 e 12 neurônio na camada intermediária e 3, na saída. Essas três unidades de saída classificam-se em padrões do tipo α-hélices, folhas-β e *coils*.

Tabela 5.7: Arquiteturas de 18 RNAs

Janelas de Resíduo de Aminoácidos	Arquitetura da 1° RNA (22 unidades)	Arquitetura da 2° RNA (25 unidades)
7	154 x 15 x 15 x 15 x 3	175 x 15 x 15 x 15 x 3
9	198 x 15 x 15 x 15 x 3	225 x 15 x 15 x 15 x 3
11	242 x 15 x 15 x 15 x 3	275 x 15 x 15 x 15 x 3
13	286 x 15 x 15 x 15 x 3	325 x 15 x 15 x 15 x 3
15	330 x 15 x 15 x 15 x 3	375 x 15 x 15 x 15 x 3
17	374 x 15 x 15 x 15 x 3	425 x 15 x 15 x 15 x 3
19	418 x 15 x 15 x 15 x 3	475 x 15 x 15 x 15 x 3
21	462 x 15 x 15 x 15 x 3	525 x 15 x 15 x 15 x 3
23	506 x 15 x 15 x 15 x 3	575 x 15 x 15 x 15 x 3

5.4.3 Implementação do Júri de Decisão

Finalmente, com a rede treinada e avaliada, ela pode ser integrada em um sistema do ambiente operacional da aplicação. Para maior eficiência da solução, este sistema deverá conter facilidades de utilização como interface conveniente e facilidades de aquisição de dados através de planilhas eletrônicas, interfaces com unidades de processamento de sinais, ou arquivos padronizados.

A inclusão do júri de decisão destina-se a fazer uma leitura de predição final (ROST & SANDER, 1993). Este júri tem a função de realizar a média aritmética sobre os resultados da predição das 18 redes. Logo após o cálculo da média, executa-se o critério de classificação. Essa classificação se baseia na análise do maior valor em cada coluna. Se a primeira coluna for o maior valor, classifica-se como α-hélice, se for a segunda, classifica-se como *coil* e se for a terceira, como folha-β, gerando, assim, o resultado da predição da estrutura secundária do resíduo em questão.

Na prática, é como se o júri recebesse a predição de diferentes redes e realizasse uma média para decidir qual a estrutura está associada ao resíduo central.

5.5 Arquitetura da Rede Neural *MLP* Otimizada (Rede Neural *oMLP*)

As Redes Neurais *MLP* representam um dos mais efetivos e extensamente métodos de aprendizado de máquina usados atualmente aplicados à classificação diagnóstica baseada em dados genômicos em alta dimensão (WANG, WANG *et al.*, 2006). Porém, como as dimensionalidades dos dados genômicos existentes frequentemente excedem os tamanhos de amostras disponíveis por ordens de magnitude, o desempenho da rede *MLP* pode diminuir gradualmente devido ao problema do aumento da dimensionalidade e do *overfitting*, e pode não oferecer uma acurácia de predição aceitável. Problemas equivalentes acontecem com os dados das sequências de aminoácidos utilizados na predição de estruturas secundárias de proteínas. Por estas razões, desenvolveu-se uma arquitetura de rede neural, denominada *oMLP* (MORAIS, 2008), com inicializações não aleatórias com o intuito de aumentar a acurácia de predição de estruturas de proteínas.

5.5.1 Considerações Iniciais

Um tipo comum de classificador de rede neural aplicado à predição de estruturas secundárias de proteínas é uma *Perceptron* Multicamadas *Feed-Forward* treinada com algoritmo *backpropagation*. Os vetores de entrada e os correspondentes vetores de saída são usados para treinar uma rede *MLP*, em um processo que atualiza os pesos e as inclinações até a rede aproximar a função de mapeamento que associa vetores de entrada com vetores de saída específicos. A propriedade de generalização torna possível treinar uma rede *MLP* com um conjunto representativo de pares entrada/saída e alcançar bons resultados na predição de proteínas jamais classificadas. A habilidade de uma rede *MLP* aprender mapeamento complexo (não-linear) e multidimensional a partir de uma coleção de exemplos faz dela um classificador ideal para predição de estruturas secundárias (HAYKIN, 1999) (KHAN *et al.*, 2001) (O'NEIL & SONG, 2003) (WEI *et al.*, 2005).

Apesar dos trabalhos bem sucedidos, reportados anteriormente, na aplicação de redes *MLP* na predição da estrutura secundária, um dos problemas mais críticos, o aumento da dimensionalidade, não tem sido resolvido eficientemente. O aumento da dimensionalidade é causado por uma quantidade finita de dados de treinamento disponível relativo a um grande espaço de características de entradas. Quando a dimensionalidade aumenta de forma considerável e a informação disponível permanece

inadequada, um grande número de parâmetros do modelo não pode ser bem treinado (HAYKIN, 1999) (JAIN *et al.*, 2000). Consequentemente, o desempenho do classificador pode diminuir gradualmente além de certo ponto, com a inclusão de características ou dimensões. Em estudos atuais, as abordagens para evitar o aumento da dimensionalidade são geralmente limitadas a diretamente reduzir o número de entradas. Existem métodos de redução de dimensionalidade convencionais, tais como: análise de componentes (KHAN *et al.*, 2001) (WEI *et al.*, 2005), *t*-estatísticas (GOLUB *et al.*, 1999), medida de correlação (VAN'T VEER *et al.*, 2002) e Rede *MLP* com treinamento baseado no procedimento de seleção de sequências, que seleciona as sequências com maior influência nas mudanças das saídas em uma *MLP* (informações evolucionárias).

Os parâmetros de projeto no treinamento de uma rede *MLP* incluem valores iniciais dos parâmetros do modelo (pesos sinápticos e inclinações), regras de parada, arquitetura da rede *MLP*, entre outros. Como não existe um algoritmo padrão para resolver este problema, a inicialização da *RNA* é feita aleatoriamente e o desempenho da classificação depende em grande parte dos valores iniciais de pesos e inclinações. Além disso, a alta complexidade dos classificadores resulta frequentemente em mínimos locais nas superfícies de erro e o treinamento dos classificadores pode ser facilmente convergido em tais mínimos locais (RAUDYS & SKURIKHINA, 1992) (RAUDYS, 1997).

Imagina-se que o desenvolvimento de uma inicialização otimizada da rede *MLP*, irá permitir a redução do aumento de dimensionalidade e assim, melhorar o desempenho da rede. O objetivo é encontrar um efetivo esquema de inicialização não aleatória que alcance um estado inicial mais próximo da solução ótima que é buscada após o treinamento (WANG, WANG *et al.*, 2004).

Esta abordagem é sustentada através de estudos prévios no campo de reconhecimento de padrão estatístico, onde foi mostrado que inicializações não-aleatórias de pesos sinápticos e inclinações de rede *MLP* resultaram em uma *MLP* com poucos erros de generalização até mesmo quando o número de amostras é menor que o número de características ou dimensões (RAUDYS, 1992) (RAUDYS, 1997) (RAUDYS & SKURIKHINA, 1992).

O esquema de inicialização não aleatória será buscado através de duas frentes, que serão detalhadas a seguir: a) Inicializações da arquitetura baseadas no Critério de Pesos de Fisher, a saber: inicialização das camadas intermediária e determinação do tamanho da camada intermediária; e b) Seleção dos dados de entrada.

5.5.2 Inicialização da Camada Intermediária

A rede *MLP* oferece um procedimento integrado para extração de padrões e classificação por aprendizado *backpropagation* (HAYKIN, 1999). Sua arquitetura auto-associativa *feed-forward* pode também ser usada para construir subespaços não-lineares no modo supervisionado e não-supervisionado (HAYKIN, 1999) (JAIN *et al.*, 2000).

A saída de uma camada intermediária pode ser interpretada como um conjunto de novos padrões apresentados à camada de saída para a classificação (HAYKIN, 1999). Por outro lado, a Análise Multiclasse Discriminante Linear (*LDA – multi-class Linear Discriminant Analysis*) oferece uma predição multivariada por estimativa de função de densidade. Seu subespaço que é extraído baseado no Critério de Pesos de Fisher (*wFC – weighted Fisher Criterion*), controlando mais eficientemente a separabilidade da classificação (LOOG, DUIN *et al.*, 2001).

Como desde a camada intermediária, já temos algum tipo de padrão, criamos em cada camada intermediária subespaços por Critério de Pesos de Fisher, objetivando controlar mais eficientemente a separabilidade da classificação. Isto pode mostrar que a determinação da Redução de Dimensão Linear (*LDR – Linear Dimension Reduction*) é equivalente a encontrar os parâmetros de Máxima Verossimilhança estimados de um modelo de Mistura Normal Finita Padrão (*SFNM – Standard Finite Normal Mixture*) (LOOG, DUIN *et al.*, 2001). Isto motivou a exploração de conexões entre rede *MLP* e Redução de Dimensão Linear.

Uma hipótese natural é que os rótulos das classes usados como vetores de saídas desejadas durante o treinamento supervisionado forçam a saída da camada intermediária a capturar a maioria dos componentes ou subespaços discriminatórios para distinguir as classes. Baseado nestas observações teóricas sugere-se uma inicialização baseada no Critério de Pesos de Fisher para a camada intermediária da rede *MLP* (WANG, WANG *et al.*, 2004). Para limitar a complexidade da rede *MLP*, assume-se que o número de neurônios da camada intermediária é menor do que o número de entradas.

Dada uma entrada m_0-dimensional de espaço t com k_0 classes, a Redução de Dimensão Linear Multiclasse procura por uma transformação linear W que transforme o espaço de entrada original para um espaço x m_1-dimensional mais baixo ($m_1 < m_0$). O espaço x extraído deve preservar a quantidade máxima de informação discriminatória da classe.

Como é muito complexo usar diretamente o erro quadrático como critério, a técnica mais comumente empregada para encontrar esta transformação é a Redução de Dimensão Linear baseada no Critério de Pesos de Fisher (JAIN et al., 2000) (HAYKIN, 1999). Este método maximiza a relação entre a matriz de espalhamento entre classes e a matriz de espalhamento dentro das classes, garantindo assim a separabilidade máxima. Neste trabalho aplica-se o Critério de Pesos de Fisher para o problema de classificação multiclasses (LOOG, DUIN *et al.*, 2001), como definido no item 2.6.5, através da equação 2.60, aqui reescrita*:*

$$J(\mathbf{w}) = Traço\left\{\left(\mathbf{W}\mathbf{S}_{\mathrm{W}}\mathbf{W}^{\mathrm{T}}\right)^{-1}\left(\mathbf{W}\mathbf{S}_{\mathrm{B}}\mathbf{W}^{\mathrm{T}}\right)\right\} \qquad \textbf{(5.1)}$$

Loog, Duin *et al.* (2001) mostraram que quando existe mais do que duas classes para ser classificadas, o Critério de Fisher Convencional Multiclasses (*cFC - conventional Fisher Criterion*) para localização de subespaços de dimensão reduzida é sub-ótimo para a respectiva classificação. A razão é que o Critério de Fisher Convencional trata pares de classes com várias distâncias iguais entre classes. Em contraste, o Critério de Pesos de Fisher incorpora uma função de peso que aproxima a taxa de erro entre as classes e designa pesos maiores para pares de classes próximos e pesos menores para pares mais distantes. O objetivo é sempre alcançar pequenas variações intra-classes e grandes variações inter-classes. Assim, no subespaço extraído encontrado pelo Critério de Pesos de Fisher, as classes com superposição alta ganham a ênfase adequada e os pares distantes permanecem separados.

Tomando somente os autovetores m_1 correspondendo aos maiores autovalores m_1 ($m_1 < m_0$), pode-se formalizar uma transformação que não reduz somente a dimensionalidade do espaço de entrada original, mas também retém a informação da classe de separabilidade máxima. Este procedimento chama-se de Análise de Componente Discriminatória baseada no Critério de Pesos de Fisher (*wFC-Discriminatory Component Analysis*).

Com a transformação $\mathbf{W}$ (m_0 x m_1) alcançada através da Redução de Dimensão Linear, o subespaço *x* característico reduzido dimensional com m_1 dimensões torna-se $\mathbf{x}_i = \mathbf{W}^T(\mathbf{t}_i - \mathbf{b}_{t0})$, para $i = 1,2,...,N$, onde N é o número de amostras, $\mathbf{x}_i$ é a representação do vetor de amostras $\mathbf{t}_i$ no espaço $\mathbf{x}$ com $x_{r,i} = \mathbf{w}_r^T(\mathbf{t}_i - \mathbf{b}_{t0})$, para $r = 1,2,...,m_i$ e $\mathbf{b}_{t0}$ é o elemento central do conjunto de dados.

Por outro lado, as saídas da camada intermediária na rede *MLP* podem ser alcançadas como $a_n = \varphi(\mathbf{w}_n^{l\,T}\mathbf{p} - b_{l,n})$, onde $\mathbf{w}_n^l$ é um conjunto de pesos sinápticos conectando m_0 entradas ao neurônio n na camada intermediária, a_n é a saída do neurônio n, $\mathbf{p}$ é o vetor de entrada da rede *MLP*, $b_{l,n}$ é o limiar do neurônio n e $\varphi(\cdot)$ é a função de ativação (HAYKIN, 1999).

A conexão entre a Redução de Dimensão Linear e o mecanismo de extração de padrões sugere que os vetores coluna da Redução de Dimensão Linear da matriz $\mathbf{W}$ podem ser usados para inicializar os pesos entre as camadas de entrada e intermediária da rede *MLP*, $\mathbf{w}_n^l = \mathbf{w}_n$, e suas inclinações podem ser inicializados como, $\mathbf{b}_l = \mathbf{W}^T\mathbf{b}_{t0}$. Os novos padrões são executados pela função de ativação $w(\cdot)$ que pode ser linear ou não-linear. Foi mostrado teoricamente que a minimização do erro em relação aos pesos sinápticos e inclinações da rede *MLP* é equivalente a maximizar o Critério de Pesos de Fisher (Equação 5.1), e isto pode ser totalmente determinado por neurônios da camada intermediária (HAYKIN, 1999).

5.5.3 Determinação do Tamanho da Camada Intermediária

O método de inicialização da rede *MLP* baseada no critério de pesos de Fisher pode também sugerir um número satisfatório de neurônios intermediários, um componente chave da arquitetura *MLP*. Redes neurais artificiais, como outros métodos flexíveis de estimação não-linear, são vulneráveis a problemas de *underfitting* e *overfitting* (HAYKIN, 1999) (RIPLEY, 1996).

O problema de *overfitting* ocorre mais facilmente quando o número de amostras do conjunto de treinamento é pequeno e a rede é relativamente grande, que é o caso da maioria dos dados proteômicos. Então é importante usar uma rede que tenha apenas o tamanho suficiente para oferecer um ajuste adequado. O subespaço resultante representado pelas saídas da camada intermediária deve manter a separabilidade da classe tanto quanto possível (HAYKIN, 1999) e a separabilidade parcial retida é dada por J_{wFC}.

Consequentemente, é apropriado deixar o número de pseudo estruturas secundárias (por exemplo, m_1, o número de neurônios intermediários) ser o número significante de autovalores alcançados pela Análise de Componente Discriminatória baseada no Critério de Pesos de Fisher, porque autovalores representam uma

separabilidade de classes no espaço característico. Neste estudo, selecionou-se um subconjunto de autovalores dominantes que contém 99% da separabilidade total e igualou-se o número de neurônios intermediários ao número de autovalores selecionados.

5.5.4 Seleção dos Dados de Entrada

As fases de treinamento e testes para classificação de estruturas de proteínas será realizada por entradas pré-selecionadas, porque a seleção da entrada é um pré-requisito fundamental para predição de estruturas secundárias devido à alta dimensionalidade do problema. Aplicou-se um recente método de dois passos de seleção de entrada baseado no Critério de Pesos de Fisher (XUAN, DONG *et al.*, 2004) que compartilha a base teórica da amostra (*wFC*) com a abordagem de inicialização da rede *MLP* proposta.

Primeiro, classifica-se todas as sequências baseadas em suas medidas discriminatórias individuais mensuradas por Critério de Pesos de Fisher unidimensional (XUAN, DONG *et al.*, 2004); uma sequência será selecionada como uma sequência discriminatória individual (*IDS* - *Individually Discriminatory Sequence*) se a sua medida de discriminação estiver acima do *threshold* empírico.

Segundo, de um conjunto de *IDS*, seleciona-se um subconjunto (de vários tamanhos) de sequência discriminatória conjunta (*JDS* - *Jointly Discriminatory Sequence*), de quem a medida discriminatória conjunta é a máxima entre todos os conjuntos de mesmo tamanho.

A medida discriminatória conjunta é também determinada por uma versão do Critério de Pesos de Fisher multidimensional (Equação 5.1). Além disso, os conjuntos *JDS* foram refinados por testes em uma rede *MLP* treinada, o qual determina um subconjunto ótimo de predição de estruturas de proteínas que minimiza o erro de generalização. A partir da curva de taxa de classificação por subconjuntos *JDS*, escolhe-se o subconjunto *JDS* ótimo que corresponde à taxa de classificação máxima como a entrada final para a rede *MLP*. Este passo melhora o desempenho da rede *MLP* e também determina o seu número de entradas m_0.

Capítulo VI – Resultados

Neste capítulo são discutidos os treinamentos, os testes e os resultados da implementação das redes *cMLP* e *oMLP*. Também se faz uma comparação direta com os principais preditores disponíveis.

6.1 Bases de Dados utilizadas

As diferentes arquiteturas de redes neurais expostas no capítulo anterior foram treinadas com quatro bases de aminoácidos diferentes:

a) ***Todas***, o qual possui todas as 106 proteínas coletadas, totalizando 32.049 resíduos, distribuídos percentualmente de acordo com a Figura 6.1

Figura 6.1: Distribuição Percentual dos Resíduos do Subconjunto de Treinamento *Todas*

b) ***Hélice***, que contém 32 proteínas, totalizando 6.622 resíduos, distribuídos percentualmente de acordo com a Figura 6.2;

c) ***Folha***, que contém 29 proteínas, totalizando 6.637 resíduos, distribuídos percentualmente de acordo com a Figura 6.3;

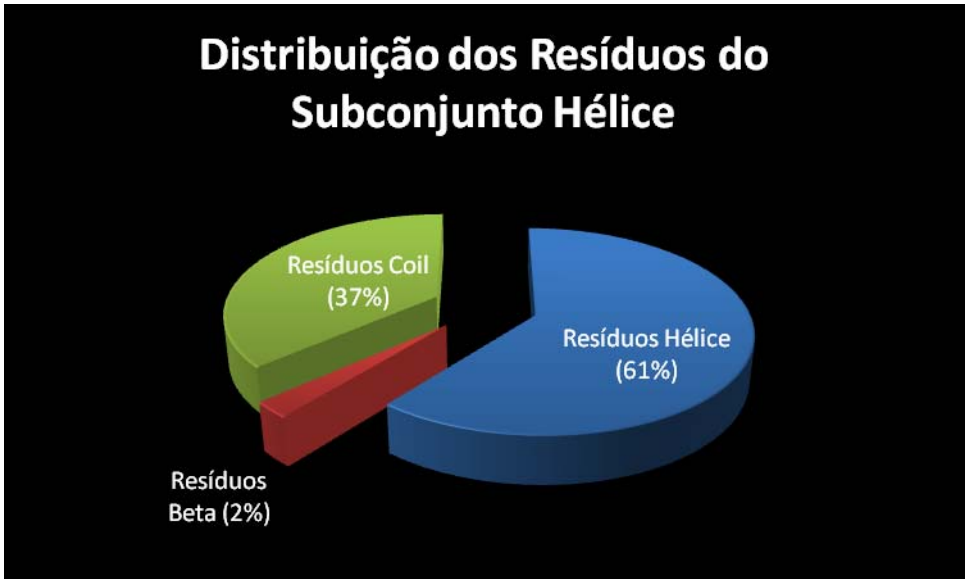

Figura 6.2: Distribuição Percentual dos Resíduos do Subconjunto de Treinamento *Hélice*

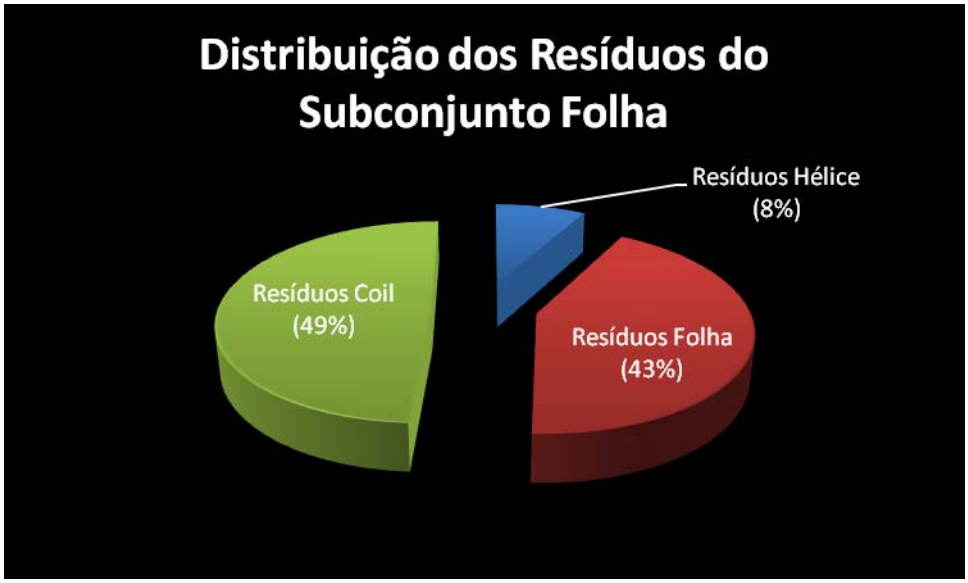

Figura 6.3: Distribuição Percentual dos Resíduos do Subconjunto de Treinamento *Folha*

d) ***Hélice-Folha***, que contém 79 proteínas, totalizando 24.681 resíduos, distribuídos percentualmente de acordo com a Figura 6.4;

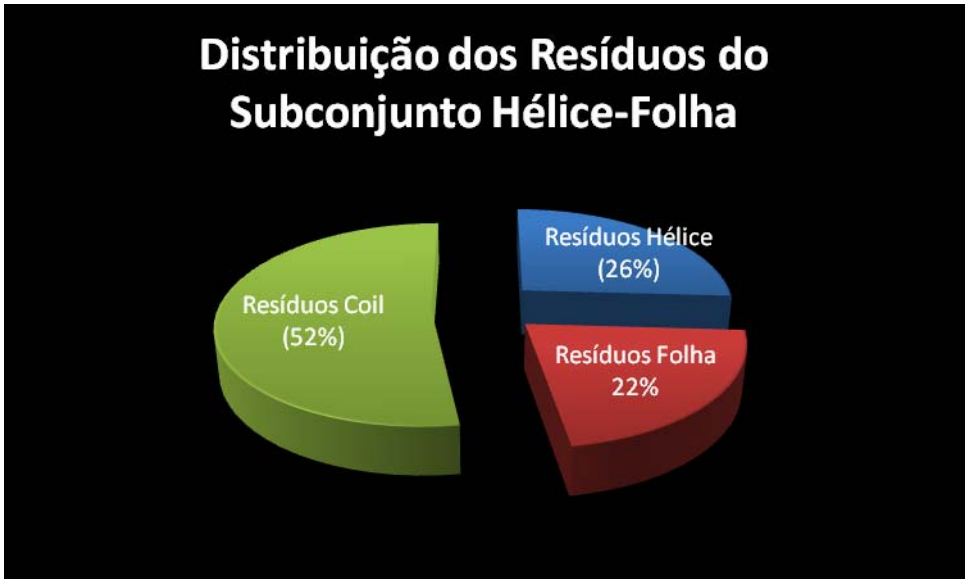

Figura 6.4: Distribuição Percentual dos Resíduos do Subconjunto de Treinamento *Hélice-Folha*

6.2 Coeficientes utilizados

Na literatura existem alguns coeficientes para verificar o desempenho de uma Rede Neural Artificial na predição de estruturas secundárias. Dentre estes, o mais utilizado pelos pesquisadores é o Q_3, que fornece a porcentagem dos resíduos preditos corretamente considerando os três tipos de estruturas secundárias: α-hélices, folhas-β e *coils*, e por este motivo, será o coeficiente utilizado neste trabalho. O coeficiente Q_3 é dado por:

$$Q_3 = \frac{\sum_{i=H,E,C} COR_i}{\sum_{i=H,E,C} TOT_i} * 100 \qquad (6.1)$$

onde:

i: {α-hélices, folhas-β, *coils*};

COR_i: número de estruturas secundárias que a RNA identificou corretamente;

TOT_i: número de estruturas secundárias existentes na proteína de teste.

Outros coeficientes importantes e que também foram utilizados na pesquisa para ajudar na avaliação dos testes são: $Q_i(obs)$ e $Q_i(pred)$. Estes coeficientes são utilizados para cada estrutura secundária em particular. O primeiro fornece a porcentagem do número de resíduos preditos corretamente em relação ao número real observado. O segundo fornece a porcentagem do número de resíduos preditos corretamente em

relação ao número que a rede neural identificou, em um estado particular. Os dois coeficientes podem ser calculados por:

$$Q_i(obs) = \frac{COR_i}{TOT_i} * 100 \qquad Q_i(pred) = \frac{COR_i}{PRED_i} * 100 \tag{6.2}$$

onde:

$PRED_i$: número de estruturas secundárias que a RNA identificou.

6.3 Resultados Obtidos com a Rede *cMLP*

Nesta seção, primeiramente, serão apresentados os resultados obtidos com o treinamento das arquiteturas com uma e duas redes neurais. Após, serão apresentados os resultados obtidos com o júri de decisão para as proteínas de teste. O objetivo destas simulações é verificar qual base de treinamento possui o melhor desempenho e qual das arquiteturas possui o melhor rendimento, para assim definir qual será a arquitetura de rede neural que representará o preditor *cMLP*. Por fim, os resultados do preditor *cMLP* serão comparados com outros preditores.

6.3.1 Treinamento da Arquitetura com uma *RNA*

Nesta arquitetura de rede neural única foi utilizada a configuração de rede que obteve melhores resultados no trabalho de Qian & Sejnowski (1988). A rede é composta de 5 camadas (entrada, 3 intermediárias e saída). A camada de entrada possui 286 neurônios (janelamento de 13 aminoácidos x 22), as camadas intermediárias possuem 15 neurônios e a saída 3 neurônios, cada um representando uma classe de estrutura secundária. A função de ativação da camada de entrada é linear e das demais camadas é sigmoide. Todas as simulações foram realizadas com um ciclo de treinamento igual a 1000. Nesta fase se calculou o erro de treinamento, isto é, as redes depois de treinadas, com respectivas bases de treinamento, foram validadas por um conjunto de 14 proteínas descritas na Tabela 5.2. Os resultados de treinamento da arquitetura com uma *RNA*, para os quatro subconjuntos de proteínas representando as bases todas, hélice, folha e hélice-folha, estão descritos na Tabela 6.1.

Tabela 6.1: Resultados obtidos na fase validação com a Arquitetura de uma *RNA*

Base de Treinamento	Q_3 (%)	Q_α(obs) (%)	Q_α(pre) (%)	Q_β(obs) (%)	Q_β(pred) (%)	Q_c(obs) (%)	Q_c(pred) (%)	Erro Trein. (%)
Todas	52,0	38	53	36	44	65	67	11,3
Hélice	52,9	66	44	18	48	58	64	7,0
Folha	46,6	7	43	72	35	42	61	1,7
Hélice-Folha	55,4	43	50	40	44	65	70	6,5

6.3.2 Treinamento da Arquitetura com duas *RNAs*

O principal objetivo foi verificar o desempenho entre os resultados obtidos com uma arquitetura composta com duas redes e aqueles obtidos anteriormente, utilizando apenas uma rede. Nesta arquitetura com duas redes mantiveram-se as proteínas de validação e a configuração da arquitetura com apenas uma rede, com a única diferença acontecendo na camada de entrada da segunda rede que passou a ter 325 neurônios, uma vez que os vetores de entrada são compostos, agora, por 25 unidades, as 22 originais e mais 3 que representam a saída da primeira rede. Os resultados de treinamento da arquitetura com duas *RNAs*, para os quatro subconjuntos de proteínas representando as bases todas, hélice, folha e hélice-folha, estão descritos na Tabela 6.2.

Tabela 6.2: Resultados obtidos na fase validação com a Arquitetura de duas *RNA*s

Base de Treinamento	Q_3 (%)	Q_α(obs) (%)	Q_α(pred) (%)	Q_β(obs) (%)	Q_β(pred) (%)	Q_c(obs) (%)	Q_c(pred) (%)	Erro Trein. (%)
Todas	52,0	38	56	47	44	62	66	3,7
Hélice	53,5	63	47	24	50	60	64	2,7
Folha	48,4	22	50	70	38	61	67	1,5
Hélice-Folha	56,2	45	49	46	45	66	66	2,4

Percebe-se com estes resultados que o erro de treinamento de arquiteturas de *RNA*s projetadas com duas redes diminui em relação ao obtido com uma única rede. Observa-se, também, que arquiteturas compostas por duas redes obtêm uma melhoria no desempenho da predição, como é amplamente divulgado na literatura (CHANDONIA & KARPLUS, 1996).

A melhor porcentagem de acerto verificada, tanto com uma, quanto com duas redes, foi com a base de treinamento hélice-folha, alcançando 55,4% para a arquitetura simples e 56,2% para a arquitetura em cascata.

6.3.3 Realização dos Testes com Arquiteturas de Uma e Duas Redes Neurais

As diferentes redes neurais projetadas foram treinadas com as quatro bases de dados diferentes e as validações mostraram que as *RN*As treinadas com a base hélice-folha, constituída de 79 proteínas não homólogas, obteve o melhor desempenho na predição de estruturas secundárias. Portanto, os resultados apresentados neste teste são das redes neurais treinadas com o subconjunto hélice-folha.

Para os testes foram utilizadas as proteínas da Tabela 5.3. Essas proteínas são usadas pelo *CASP* na avaliação dos métodos de predição de estrutura secundária. Este conjunto de proteínas totaliza 1.095 resíduos distribuídos percentualmente como ilustra a Figura 6.5.

Figura 6.5: Distribuição Percentual dos Resíduos das Proteínas de Teste

Com o objetivo de verificar a porcentagem de acerto de cada janela foram realizados testes com nove redes com janelas variando de 7 a 23. Primeiramente as projetadas com uma rede e depois as projetadas com duas redes neurais, totalizando 18 redes neurais. As Tabelas 6.3 e 6.4 apresentam tais resultados.

Tabela 6.3: Porcentagem de acerto dos testes da Arquitetura de uma *RNA*

Proteína	7 Jan. (%)	9 Jan. (%)	11 Jan. (%)	13 Jan. (%)	15 Jan. (%)	17 Jan. (%)	19 Jan. (%)	21 Jan. (%)	23 Jan. (%)	Média dos acertos por Proteína
1QLQ	68	59	58	47	21	58	47	60	58	**52,89**
1EIG	51	58	53	49	37	54	61	56	58	**53,00**
1C56	46	57	52	51	59	57	62	49	58	**54,56**
1DAQ	45	64	44	55	58	49	38	59	60	**52,44**
1EHD	58	49	55	65	55	55	57	49	48	**54,56**
1E5B	59	52	52	47	58	57	49	53	58	**53,89**
1EJG	47	55	44	35	58	61	49	55	57	**51,22**
1ES1	55	60	55	61	51	54	59	57	33	**53,89**
1DT4	61	57	39	37	43	57	57	51	56	**50,89**
1EDS	50	49	42	60	54	56	52	64	51	**53,11**
1G6X	66	53	44	55	57	58	59	59	59	**56,67**
1DOI	64	64	55	54	56	65	67	61	60	**60,67**
1FD8	62	52	49	50	56	55	58	49	58	**54,33**
1FE5	61	62	61	69	55	61	50	60	57	**59,56**
1EHJ	49	61	50	44	52	59	53	57	53	**53,11**
Média dos acertos por Janelamento	**56,13**	**56,80**	**50,20**	**51,93**	**51,33**	**57,07**	**54,53**	**55,93**	**54,93**	

Tabela 6.4: Porcentagem de acerto dos testes da Arquitetura de duas *RNA*'s

Proteína	7 Jan. (%)	9 Jan. (%)	11 Jan. (%)	13 Jan. (%)	15 Jan. (%)	17 Jan. (%)	19 Jan. (%)	21 Jan. (%)	23 Jan. (%)	Média dos acertos por Proteína
1QLQ	61	60	65	43	58	67	43	65	44	**56,22**
1EIG	57	50	60	52	33	49	54	55	56	**51,78**
1C56	61	62	23	67	67	67	34	50	58	**54,33**
1DAQ	55	73	45	65	68	61	42	67	55	**59,00**
1EHD	59	57	56	65	67	65	50	56	55	**58,89**
1E5B	58	59	57	59	51	68	50	65	68	**59,44**
1EJG	47	61	49	60	59	61	44	65	59	**56,11**
1ES1	57	43	61	58	61	51	59	69	58	**57,44**
1DT4	57	61	69	59	43	62	55	51	59	**57,33**
1EDS	60	48	40	61	51	66	50	62	70	**56,44**
1G6X	65	57	54	56	57	62	53	71	68	**60,33**
1DOI	61	66	65	51	56	63	64	66	63	**61,67**
1FD8	65	55	34	53	44	59	59	55	67	**54,56**
1FE5	60	63	62	70	70	51	61	61	67	**62,78**
1EHJ	59	60	50	41	34	63	50	58	60	**52,78**
Média dos acertos por Janelamento	**58,80**	**58,33**	**52,67**	**57,33**	**54,60**	**61,00**	**51,20**	**61,07**	**60,47**	

6.3.4 Realização dos Testes com utilização do Júri de Decisão

Este resultado foi alcançado utilizando a ideia que possibilitou a melhoria de predição das redes neurais do trabalho de Rost & Sander (1993), a utilização de júri de decisão. Foram utilizados três júris: o primeiro avaliou os resultados alcançados pelas nove arquiteturas de uma rede neural (JÚRI_SIMPLES); o segundo avaliou os resultados das outras nove arquiteturas de duas redes neurais (JÚRI_DUPLO); e um terceiro que avaliava os resultados de todas as dezoito arquiteturas de rede neurais (JÚRI_FINAL). Os resultados alcançados pelos júris de decisão podem ser visualizados na Tabela 6.5.

Tabela 6.5: Resultados de predição alcançados pelos três Júris de Decisão

Proteína	JÚRI_SIMPLES (%)	JÚRI_DUPLO (%)	JÚRI_FINAL (%)	Média dos acertos por Proteína
1QLQ	62	59	65	**62,00**
1EIG	57	60	60	**59,00**
1C56	63	62	64	**63,00**
1DAQ	58	70	71	**66,33**
1EHD	66	67	70	**67,67**
1E5B	60	65	67	**64,00**
1EJG	57	71	71	**66,33**
1ES1	61	67	69	**65,67**
1DT4	66	61	71	**66,00**
1EDS	62	62	69	**64,33**
1G6X	66	65	69	**66,67**
1DOI	73	70	74	**72,33**
1FD8	60	67	69	**65,33**
1FE5	70	68	72	**70,00**
1EHJ	78	70	79	**75,67**
Média dos acertos por Tipo de Júri	**63,93**	**65,60**	**69,33**	

Os resultados apresentados na Tabela 6.5 mostram que a aplicação do júri de decisão nas nove redes projetadas de uma única rede comprovou um acerto significativo entre 57 e 78%. Na implementação do júri para a arquitetura com duas redes a acurácia alcançou o intervalo entre 59 e 71%, porém com uma média de acerto para o conjunto de proteínas de 65,60%, isto é, cerca de 1,5% melhor do que o JÚRI_SIMPLES. Na implementação do júri para as 18 arquiteturas de redes a acurácia aumentou para o

intervalo de 60 a 79%, tendo uma média de 69,33% para as 15 proteínas do conjunto de teste.

Através deste resultado, considera-se que a rede *cMLP* (rede neural perceptron de múltiplas camadas convencional), que será utilizada para comparações com outros preditores, inclusive com a rede *oMLP*, terá a seguinte configuração: uma arquitetura composta por nove redes neurais simples e nove redes neurais em cascata (representando os janelamentos entre 7 e 23), com uma camada de júri de decisão composto pela votação das dezoito redes neurais em questão e com treinamento realizado pela subconjunto hélice-folha de proteínas (MORAIS, 2009) (MORAIS, MONDAINI & VILELA, 2009).

6.3.5 Comparação dos Resultados com outros Preditores

Os resultados obtidos com a rede *cMLP*, para as 15 proteínas de teste, foram comparados com preditores disponíveis na *Web*, para avaliar a qualidade do preditor desenvolvido. Os resultados da predição realizada pelas ferramentas para cada proteína seguem na Tabela 6.6 e na Figura 6.6.

Tabela 6.6: Comparação dos resultados da *cMLP* com os principais preditores

Proteína	cMLP (%)	PredictProtein (%)	PSIPRED (%)	JPred (%)	PREDATOR (%)	PSA (%)	PREDCASA (%)
1QLQ	65	91	87	90	69	51	84
1EIG	60	86	91	91	40	75	73
1C56	64	67	50	57	47	37	47
1DAQ	71	70	78	62	61	66	85
1EHD	70	55	59	82	53	58	52
1E5B	67	65	72	65	42	63	60
1EJG	71	50	67	72	56	58	80
1ES1	69	74	78	65	49	56	70
1DT4	71	71	78	49	48	63	64
1EDS	69	29	38	70	57	41	61
1G6X	69	91	91	80	53	53	84
1DOI	74	60	66	34	55	54	61
1FD8	69	79	84	90	49	27	79
1FE5	72	66	86	67	60	67	63
1EHJ	79	53	76	79	67	66	74
Média dos acertos por Ferramenta	**69,33**	**67,13**	**73,40**	**70,20**	**53,73**	**55,67**	**69,13**

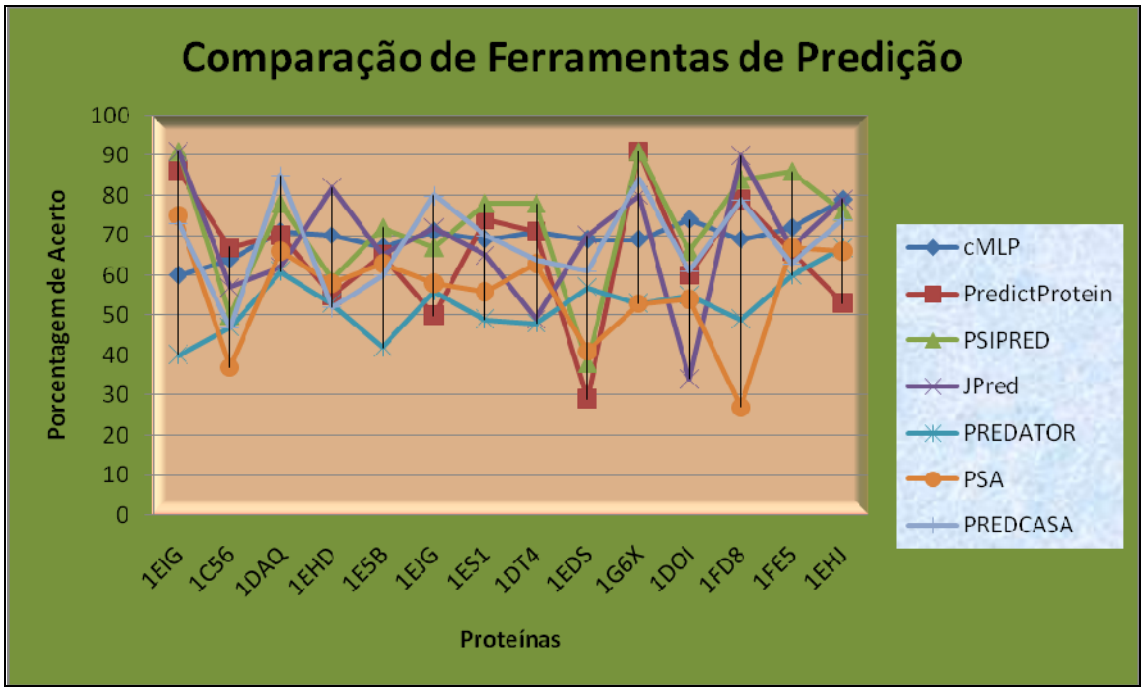

Figura 6.6: Comparação Gráfica dos resultados da *cMLP* com os principais preditores

A rede *cMLP* teve um desempenho médio melhor em relação à quatro preditores e pior em relação à dois. Para verificar que a diferença não chega a ser significativa, na Figura 6.7 a rede *cMLP* é comparada somente com os preditores *PSIPRED* e *Jpred*, isto é, somente com os preditores que apresentaram melhor desempenho. Deve-se observar que apesar da média inferior, a rede *cMLP* obteve uma função mais próxima de uma constante, não apresentando grandes diferenças de porcentagem de predição na variação das proteínas.

Também foram analisados os acertos individualizados em relação às três estruturas secundárias para que houvesse um melhor entendimento da influência da base de treinamento em relação às predições. Verifica-se que alguns preditores erram mais em algumas estruturas do que outras e isto pode ser verificado na Tabela 6.7. Foi somado o total de resíduos em formação hélice, folha e *coil* das proteínas *CASP* de teste e o total previsto pelos preditores.

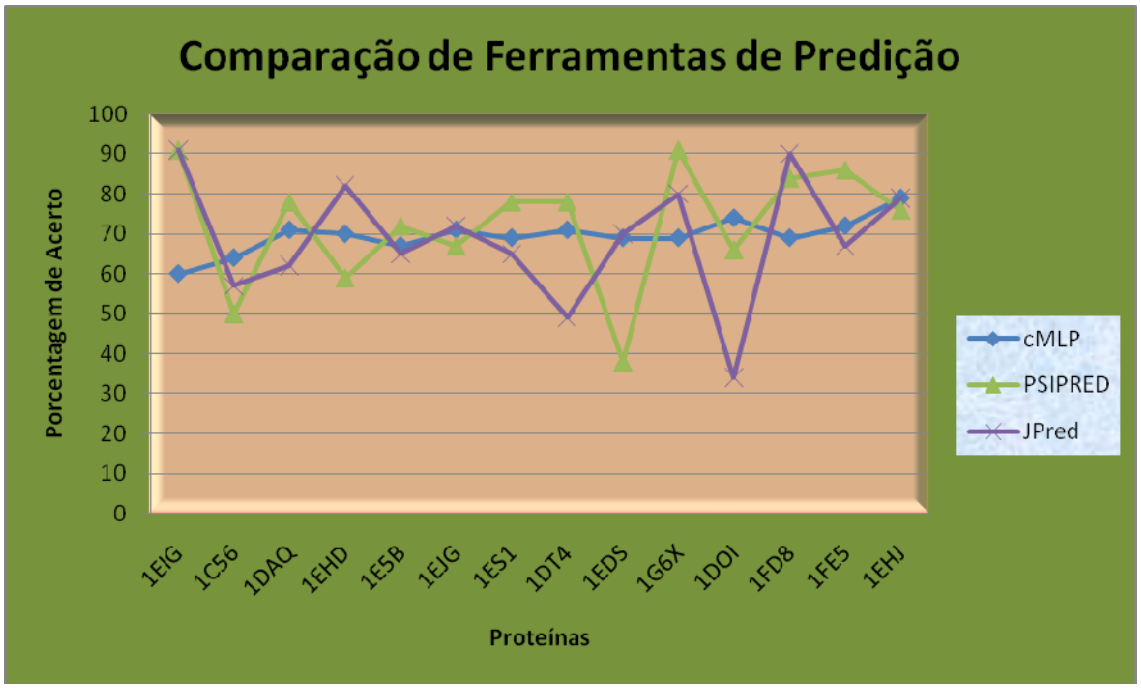

Figura 6.7: Comparação Gráfica dos resultados da *cMLP* com os preditores *PSIPRED* e *JPred*

Tabela 6.7: Acerto individualizado por estruturas secundárias dos preditores

Estrutura Secundária	Estruturas de Teste	cMLP	PredictProtein	PSIPRED	Jpred	PREDATOR	PSA	PREDCASA
α-Hélice	318	295	205	303	297	288	300	287
Folha-β	220	178	278	208	198	151	124	181
Coil	557	622	612	584	600	656	671	627

O objetivo central foi verificar o quanto a *cMLP* se aproxima em termos de acurácia aos principais preditores e assim partir para um processo de otimização do preditor que já é equivalente aos atuais, resultados de alguns trabalhos científicos.

6.4 Resultados Obtidos com a Rede *oMLP*

6.4.1 Configuração Inicial Não Aleatória da Rede Neural

Nesta arquitetura de rede neural otimizada utilizou-se uma configuração de rede alcançada pelo Critério de Pesos de Fisher. Primeiramente, classificaram-se todas as sequências do conjunto de treinamento Hélice-Folha (pelo motivo de ter sido o que obteve melhores resultados) baseadas em suas medidas discriminatórias individuais (*IDS*) através do Critério de Pesos de Fisher. Deste primeiro conjunto, selecionou-se um

subconjunto de sequência discriminatória conjunta (*JDS*) e entende-se que o conjunto *JDS* é o subconjunto ótimo de predição de estruturas de proteínas, pois minimiza o erro de generalização. Tal subconjunto é apresentado na Tabela 6.8.

Tabela 6.8: Proteínas utilizadas para treinamento para a Rede *oMLP*

				Classificação DSSP						
Proteína	Cadeias	Clas. SCOP	Clas. CATH	Total de Res.	Res. Alpha	Qtd. Hélices	% Alpha	Res. Beta	Qtd. Folhas	%Beta
1ACX	1	ALL BETA	MAINLY BETA	108	0	0	0	47	10	44
2AZAa	1 DE 2	ALL BETA	MAINLY BETA	129	21	4	16	46	11	36
2AZAb	2 DE 2	ALL BETA	MAINLY BETA	129	22	5	17	44	11	34
1AZU	1	ALL BETA	MAINLY BETA	128	14	2	11	35	10	27
1BP2	1	ALL ALPHA	MAINLY ALPHA	123	60	7	49	11	5	9
1CA2	1	ALL BETA	MAINLY BETA	259	42	10	16	77	18	30
1CCR	1	ALL ALPHA	MAINLY ALPHA	111	47	6	42	2	2	2
5CPV	1	ALL ALPHA	MAINLY ALPHA	108	61	8	56	4	2	4
1CYCa	1 DE 2	ALL ALPHA	MAINLY ALPHA	103	35	3	34	2	2	2
1CYCb	2 DE 2	ALL ALPHA	MAINLY ALPHA	103	35	3	34	2	2	2
1EST	1	ALL BETA	MAINLY BETA	240	25	6	10	94	26	39
1FC2d	2 DE 2	ALL BETA	MAINLY BETA	224	18	4	8	95	19	42
4GCR	1	ALL BETA	MAINLY BETA	174	16	4	9	84	18	48
1HDSa	1 DE 4	ALL ALPHA	MAINLY ALPHA	141	81	9	57	1	1	1
1HDSb	2 DE 4	ALL ALPHA	MAINLY ALPHA	145	80	8	55	0	0	0
1HDSd	4 DE 4	ALL ALPHA	MAINLY ALPHA	145	79	11	54	0	0	0
2IG2h	1 DE 2	ALL BETA	MAINLY BETA	455	16	5	4	104	25	23
2IG2l	2 DE 2	ALL BETA	MAINLY BETA	216	14	3	6	95	23	44
1LZ1	1	A+B	MAINLY ALPHA	130	51	7	39	16	9	12
1LZT	1	A+B	MAINLY ALPHA	129	55	7	43	14	9	11
1P2P	1	ALL ALPHA	MAINLY ALPHA	124	54	6	44	9	5	7
1PFC	1	ALL BETA	MAINLY BETA	113	4	1	4	35	7	31
1PPD	1	A+B	ALPHA BETA	212	56	7	26	45	17	21
1PYPa	1 DE 2	ALL BETA	ALPHA BETA	285	42	6	15	36	13	13
1PYPb	2 DE 2	ALL BETA	ALPHA BETA	285	42	6	15	36	13	13
1REIa	1 DE 2	ALL BETA	MAINLY BETA	107	3	1	3	52	11	49
1REIb	2 DE 2	ALL BETA	MAINLY BETA	107	3	1	3	54	11	50
1RHD	1	A/B	ALPHA BETA	293	87	12	30	39	17	13
1TIMa	1 DE 2	A/B	ALPHA BETA	247	113	13	46	44	10	18
1TIMb	2 DE 2	A/B	ALPHA BETA	247	114	12	46	43	10	17
1TGSz	2 DE 2	ALL BETA	MAINLY BETA	229	23	4	10	90	18	39
2ACT	1	A+B	ALPHA BETA	220	66	9	30	49	17	22
2CYP	1	ALL ALPHA	MAINLY ALPHA	294	147	18	50	22	11	7
2KAIb	2 DE 3	ALL BETA	MAINLY BETA	152	18	3	12	43	9	28
2KAIi	3 DE 3	ALL BETA	MAINLY BETA	58	11	2	19	12	4	21
4MDHa	1 DE 2	A/B	ALPHA BETA	333	142	14	43	64	15	19

4MDHb	2 DE 2	A/B	ALPHA BETA	333	139	13	42	61	15	18
2SBT	1	A/B	ALPHA BETA	275	59	7	21	38	10	14
2SNS	1	ALL BETA	MAINLY BETA	149	29	4	19	32	12	21
2SODb	1 DE 4	ALL BETA	MAINLY BETA	151	0	0	0	55	11	36
2SODg	1 DE 4	ALL BETA	MAINLY BETA	151	4	1	3	58	12	38
2SODo	1 DE 4	ALL BETA	MAINLY BETA	151	3	1	2	64	15	42
2SODy	1 DE 4	ALL BETA	MAINLY BETA	151	4	1	3	59	12	39
2TBVa	1 DE 3	ALL BETA	MAINLY BETA	387	7	2	2	114	26	29
2TBVb	2 DE 3	ALL BETA	MAINLY BETA	387	10	3	3	117	25	30
2TBVc	3 DE 3	ALL BETA	MAINLY BETA	387	7	2	2	116	25	30
3PGK	1	A/B	ALPHA BETA	415	143	14	34	48	16	12
3RP2a	1 DE 2	ALL BETA	MAINLY BETA	224	18	3	8	89	21	40
3RP2b	2 DE 2	ALL BETA	MAINLY BETA	224	18	3	8	82	20	37
8TLNe	2 DE 2	A/B	ALPHA BETA	316	131	12	41	54	17	17
4SBVa	1 DE 3	ALL BETA	MAINLY BETA	260	30	6	12	73	12	28
4SBVb	2 DE 3	ALL BETA	MAINLY BETA	260	44	10	17	74	11	28
4SBVc	3 DE 3	ALL BETA	MAINLY BETA	260	38	8	15	74	12	28
5AT1a	1 DE 4	A/B	ALPHA BETA	310	118	14	38	49	14	16
5CPA	1	A/B	ALPHA BETA	307	117	11	38	52	10	17
5LDHa	1 DE 2	A/B	ALPHA BETA	333	130	12	39	34	13	10
5LDHb	2 DE 2	A/B	ALPHA BETA	333	130	12	39	34	13	10
6ADHa	1 DE 2	A/B	ALPHA BETA	374	67	13	18	81	25	22
6ADHb	2 DE 2	A/B	ALPHA BETA	374	66	12	18	75	24	20
8APIa	1 DE 2	A/B	ALPHA BETA	347	103	10	30	117	12	34
8CATa	1 DE 2	MULTIDOMAIN	-	506	162	22	32	84	19	17
8CATb	2 DE 2	MULTIDOMAIN	-	506	158	21	31	84	19	17
1LH1	1	ALL ALPHA	MAINLY ALPHA	153	119	9	78	0	0	0

Este novo conjunto de treinamento contém 42 proteínas, totalizando 14.630 resíduos, sendo 3.571 hélices e 3.264 folhas. Pela primeira vez os pesos sinápticos não foram inicializados com valores aleatórios. Os valores das conexões foram extraídos da Equação 5.1, que prevê a distribuição de pesos sinápticos de acordo com o conjunto de entrada da rede neural.

A arquitetura da rede é composta de 5 camadas, sendo 1 de entrada, 3 intermediárias e 1 de saída, assim como também era a rede *cMLP*. Nesta rede selecionou-se um subconjunto de autovalores dominantes (do conjunto de treinamento Hélice-Folha) que contém 99% da separabilidade total e igualou-se o número de neurônios intermediários ao número de autovalores selecionados e assim, se alcançou o número de camadas intermediárias para este conjunto de treinamento, uma vez que cada camada continua a conter 15 neurônios. A função de ativação da camada de entrada é

linear e das demais camadas é sigmoide. Todas as simulações foram realizadas com um ciclo de treinamento igual a 1000.

6.4.2 Treinamento da Arquitetura Otimizada

Os resultados de treinamento das arquiteturas com uma *RNA* e duas *RNAs*, para o conjunto de proteínas descrito na Tabela 6.8, são apresentados na Tabela 6.9

Tabela 6.9: Resultados obtidos na fase treinamento com as arquiteturas otimizadas de uma e duas *RNAs*

Arquitetura de Rede	Q_3 (%)	Q_α(obs) (%)	Q_α(pre) (%)	Q_β(obs) (%)	Q_β(pred) (%)	Q_c(obs) (%)	Q_c(pred) (%)	Erro Trein. (%)
Com uma RNA	64,2	55	57	53	50	69	72	4,5
Com duas RNAs	66,1	66	61	49	67	69	71	1,4

Em ambos os casos, tanto com uma ou com duas redes neurais, os resultados de predição melhoraram em relação à inicialização aleatória, sendo que a arquitetura com duas redes tem um melhor desempenho na predição de hélices e em relação às outras estruturas secundárias, tem um desempenho equivalente à arquitetura com apenas uma rede neural. Em relação aos erros de treinamento, a arquitetura com duas redes neurais apresentou o melhor resultado em relação a todos os treinamentos executados neste trabalho, como era esperado devido ao processo de otimização da rede neural amplamente divulgado na literatura (WANG, WANG *et al.*, 2006).

6.4.3 Realização dos Testes da Arquitetura Otimizada

Para a realização dos testes da rede *oMLP* utilizamos as mesmas proteínas da Tabela 5.3. Aproveitando os resultados da rede *cMLP*, testou-se três configurações de arquiteturas, todas utilizando o recurso de júri de decisão (ROST & SANDER, 1993) e são elas a saber: JÚRI_SIMPLES (resultados das 9 arquiteturas com uma rede neural), JÚRI_DUPLO (resultados das 9 arquiteturas com duas redes neurais) e JÚRI_FINAL (resultados das 18 arquiteturas de rede neural). Os resultados alcançados pelos júris de decisão podem ser visualizados na Tabela 6.10.

Os resultados apresentados na Tabela 6.10 mostram que a inicialização não aleatória da configuração da rede neural representa um fator significativo para a melhoria do desempenho de predição. No melhor caso, que foi a implementação de um

júri de decisão que computava os resultados de todas as 18 redes neurais, obteve-se uma média de acerto de 74,93% para as 15 proteínas de teste.

Tabela 6.10: Resultados de predição alcançados pelos três Júris de Decisão para a arquitetura *oMLP*

Proteína	JÚRI_SIMPLES (%)	JÚRI_DUPLO (%)	JÚRI_FINAL (%)	Média dos acertos por Proteína
1QLQ	66	67	70	**67,67**
1EIG	59	65	67	**63,66**
1C56	65	66	72	**67,67**
1DAQ	67	63	77	**66,33**
1EHD	65	72	72	**69,00**
1E5B	67	70	74	**70,33**
1EJG	73	74	72	**73,00**
1ES1	64	65	66	**65,00**
1DT4	67	62	79	**69,33**
1EDS	61	67	82	**70,00**
1G6X	71	69	80	**73,33**
1DOI	74	75	85	**78,00**
1FD8	63	78	71	**70,67**
1FE5	72	70	75	**72,33**
1EHJ	75	75	82	**77,33**
Média dos acertos por Tipo de Júri	**67,26**	**69,20**	**74,93**	

Através deste resultado, considera-se que a rede *oMLP* (rede neural perceptron de múltiplas camadas otimizada), que será utilizada para comparações com outros preditores, inclusive com a rede *cMLP*, terá a seguinte configuração: uma arquitetura composta por nove redes neurais simples e nove redes neurais em cascata (representando os janelamentos entre 7 e 23), com uma camada de júri de decisão composto pela votação das dezoito redes neurais em questão, inicializando com pesos calculados pelo Critério de Pesos de Fisher e com treinamento realizado pela subconjunto hélice-folha otimizado pelo Critério de Pesos de Fisher.

6.4.4 Comparação dos Resultados com outros Preditores

Os resultados obtidos com a rede *oMLP*, para as 15 proteínas de teste, foram comparados com preditores disponíveis na *Web*, para avaliar a qualidade do preditor

desenvolvido. Os resultados da predição realizada pelas ferramentas para cada proteína seguem na Tabela 6.11

Tabela 6.11: Comparação dos resultados da *oMLP* com os principais preditores

Proteína	oMLP (%)	cMLP (%)	PredictProtein (%)	PSIPRED (%)	JPred (%)	PREDATOR (%)	PSA (%)	PREDCASA (%)
1QLQ	70	65	91	87	90	69	51	84
1EIG	67	60	86	91	91	40	75	73
1C56	72	64	67	50	57	47	37	47
1DAQ	77	71	70	78	62	61	66	85
1EHD	72	70	55	59	82	53	58	52
1E5B	74	67	65	72	65	42	63	60
1EJG	72	71	50	67	72	56	58	80
1ES1	66	69	74	78	65	49	56	70
1DT4	79	71	71	78	49	48	63	64
1EDS	82	69	29	38	70	57	41	61
1G6X	80	69	91	91	80	53	53	84
1DOI	85	74	60	66	34	55	54	61
1FD8	71	69	79	84	90	49	27	79
1FE5	75	72	66	86	67	60	67	63
1EHJ	82	79	53	76	79	67	66	74
Média dos acertos por Janelamento	**74,93**	**69,33**	**67,13**	**73,40**	**70,20**	**53,73**	**55,67**	**69,13**

Tomando como métrica o parâmetro Q_3, a rede *oMLP* teve um desempenho médio melhor em relação aos preditores avaliados, porém sem grandes ganhos significativos. Para comprovar tal ideia, na Figura 6.8 a rede *oMLP* é comparada somente com os preditores *cMLP, PSIPRED* e *Jpred*, isto é, com a rede com inicialização aleatória e apenas com os preditores que apresentaram melhor desempenho em relação à rede *cMLP*.

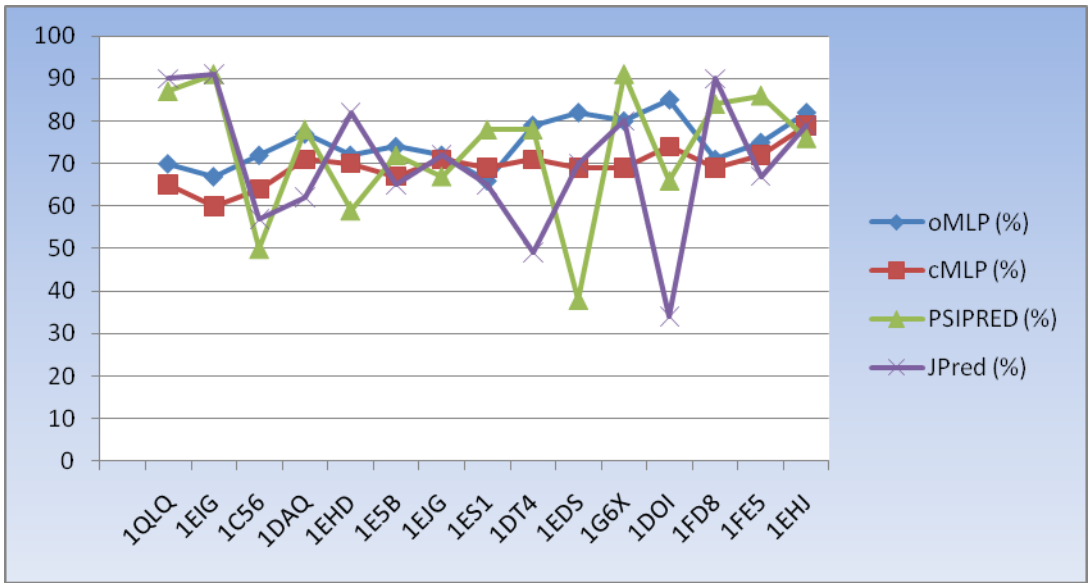

Figura 6.8: Comparação Gráfica dos resultados da Arquitetura *oMLP* com os principais preditores

Capítulo VII – Conclusões

Abordagens de Aprendizado de Máquina têm um papel fundamental na Biologia Molecular, devido a abundância de dados altamente variados e à ausência de teorias a um nível molecular. O presente trabalho apresentou um experimento utilizando redes neurais artificiais do tipo *MLP* com algoritmo de aprendizado *backpropagation* no reconhecimento de padrões de estruturas secundárias de proteínas. O problema escolhido utiliza, em princípio, a topologia e procedimentos que podem ser reproduzidos em outros tipos de problemas, com o reconhecimento de caracteres e problemas classificação de padrões em geral.

A escolha do algoritmo de aprendizado *backpropagation* se baseia nas características por ele apresentado. Pelo fato da rede neural possuir no mínimo uma camada intermediária, a mesma pode ser utilizada para solucionar diversos tipos de problemas complexos, possui maior flexibilidade, apresenta grande simplicidade de entendimento e implementação.

Embora se tenha adotado uma rede altamente flexível e poderosa, não foi encontrado na literatura consultada um trabalho amplo e prático que classificasse as redes neurais quanto a sua adequabilidade a determinado tipo de problema. Portanto, não há como avaliar se outros tipos de rede poderiam obter melhores resultados e menor esforço computacional na aplicação escolhida. Caso exista a possibilidade de comparação, somente podemos experimentar avaliar os resultados, o que não foi o foco do trabalho.

Como atualmente existe uma diversidade de programas para predição de estruturas secundárias, a ideia central foi desenvolver um preditor e comparar seus resultados com os resultados obtidos por estes preditores. A partir da construção de uma ferramenta que seja equivalente ao estado da arte na área de predição de estruturas secundárias (rede *cMLP*), partiu-se para o processo de otimização da rede com o objetivo de aumentar a acurácia da predição (rede *oMLP*).

Em linhas gerais, a rede *cMLP* processa as informações de entrada em três níveis: nível 1 (sequência-estrutura); nível 2 (estrutura-estrutura); e nível 3 (júri de decisão). No primeiro, encontram-se redes que predizem a estrutura secundária a partir da sequência primária. Essas redes, do tipo *MLP*, são não-recorrentes e compostas de

três camadas (entrada, intermediária e saída). No segundo, tem-se a implementação de uma segunda rede, em que os dados de saída da primeira são reaproveitados como entrada para a segunda. O último nível realiza uma média aritmética dos resultados da predição, obtidos de 18 redes distintas e treinadas independentemente. Na prática, é como se o júri recebesse a predição de diferentes redes e realizasse uma média para decidir qual estrutura secundária está associada para o resíduo central.

Os resultados da rede *cMLP* se equipararam em termos de acurácia de predição em relação às ferramentas consultadas, a saber: PredictProtein, PSIPRED, JPred, PREDATOR, PSA e PREDCASA.

Notou-se durante este trabalho uma enorme carência de uma metodologia consolidada para o desenvolvimento das redes neurais artificiais. Embora grandes avanços tenham sido conseguidos ao longo dos anos, não se formalizou o processo de desenvolvimento, pois não há regras que regem a determinação do número de camadas e neurônios de uma rede. Portanto, as redes *cMLP* apresentadas podem ser consideradas como experimentos realizados empiricamente de modo a balancear o número de neurônios e a precisão da rede.

Após este trabalho inicial, otimizou-se a rede neural convencional com inicializações aleatórias, para torná-la uma rede neural *MLP* otimizada (*oMLP*), e assim, realizar novos treinamentos e testes com as mesmas proteínas já treinadas anteriormente. A realização desta otimização aconteceu na utilização de parâmetros do projeto da rede neural baseada em predições por estimativa usando análise multiclasse discriminante linear e extração de subespaços com o Critério de Fisher de Pesos, inclusive na seleção de dados de entrada. Portanto, para solucionar o problema de predição de estruturas secundárias utilizando redes neurais artificiais, projetou-se e desenvolveu-se, baseado na Análise Discriminante Linear de Fisher, um esquema de rede *MLP* de duas camadas que efetivamente otimiza os parâmetros de inicialização e a arquitetura da rede *MLP*.

Os experimentos foram projetados para mostrar o impacto do método de otimização da rede *MLP* proposto sobre os dois maiores aspectos de desempenho de uma rede *MLP*: acurácia de predição e eficiência de treinamento. Esperava-se que a rede *MLP* otimizada demonstrasse consistentemente sua habilidade, aliviando o aumento da dimensionalidade em conjuntos extensos de dados de proteínas. Em comparação com uma rede *MLP* convencional, que utiliza inicialização aleatória, esperava-se obter melhorias significativas na maioria das medidas de desempenho, incluindo acurácia da

predição da estrutura secundária e propriedades de convergência. Para fins de comparação, as redes *oMLP* e *cMLP* foram treinadas e testadas com o mesmo conjunto de proteínas.

Tomando como métrica o coeficiente Q_3, podemos perceber, através da acurácia da predição, um melhor desempenho das redes com inicialização não aleatória, mas não com o ganho substancial que se esperava. O que se percebeu é que mesmo com um trabalho extenso de otimização de parâmetros apenas se conseguiu igualar percentuais de acurácia já descritos na literatura, como foi apresentado na Tabela 6.11.

Durante a execução do trabalho, avaliaram-se, também, outras técnicas de aprendizado de máquina, tais como: algoritmos genéticos, máquinas de vetor de suporte, lógica fuzzy, entre outros e percebeu-se que os melhores resultados foram alcançados por Redes Neurais para o problema de Predição de Estruturas Secundárias. Percebeu-se, também, que outros autores, tentaram juntar técnicas de aprendizado de máquina na tentativa de melhores resultados e obtiveram ganhos mínimos.

Isto pode significar que o paradigma de Redes Neurais pode ter alcançado o seu limiar em relação à predição de estruturas secundárias, isto é, entende-se que para se melhorar a acurácia de predição não se deve insistir na melhoria do modelo e sim, tentar desenvolver um novo modelo.

Uma possibilidade de continuação de trabalho seria avaliar as Redes Neurais Quânticas (ALTAISKY, 2001) (GRALEWICZ, 2004), uma vez que temos experiências anteriores de sucesso na utilização de algoritmos quânticos como são os casos da transformada de Fourier quântica (e os consequentes algoritmos de Shor e o algoritmo de Deutsch-Jozsa) e busca de Groover, e isto habilita tal modelo a uma possível melhoria no processo de predição de estruturas secundárias de proteínas.

Outra possibilidade seria aplicar o modelo de redes neurais com otimização aqui apresentado em outros problemas de Biologia Computacional, que ainda carecem de experimentos para suas respectivas soluções, tais como: reconhecimento de genes em sequências de nucleotídeos, em particular a identificação de sítios de *splice* alternativos; reconhecimento de promotores; previsão de operons em sequências de genes; reconhecimento de genes em sequências de DNAs; entre outros.

Referências Bibliográficas

ALTAISKY, M. V., 2001, Quantum neural network. Technical report, http://arxiv.org/PS_cache/quant-ph/pdf/0107/0107012.pdf.

ALTSCHUL, S.; MADDEN, T.; SHAFFER, A. *et al*., 1997, "Gapped blast and psi-blast: a new generation of protein database search programs". *Nucleic Acids Research*, v.25, n.17 (Sep), pp. 3389-3402.

BALDI, P.; BRUNAK, S.; FRASCONI, P. *et al*., 1999, "Exploiting the past and the future in protein secondary structure prediction". *Bioinformatics*, v.15, n.11 (Nov), pp. 937-946.

BALDI, P. & BRUNAK, S. 2001, *Bioinformatics – The Machine Learning Approach. Second Edition*. MIT Press, Cambridge.

BARRERA, J.; TERADA, R., JR, R.H. & HIRATA, N. S. T., 2000, "Automatic programming of morphological machines by pac learning". *Fundamenta Informaticae*, v. 41, n.1-2, pp. 229-258.

BATTITI, R., 1991, *First and second-order methods for learning: between steepest descent and Newton's method*. Technical report, University of Trento.

BENSON, G. & WATERMAN, M. S., 1994, "A method for fast database search for all k-nucleotide repeat". *Nucleic Acids Research*, v.22, n.22 (Nov), pp. 4828-4836.

BERMAN, H.; WESTBROOK, J.; FENG, Z. *et al.* 2000, "The Protein Data Bank". *Nucleic Acids Research*, v.28, n.1 (Aug), pp. 235-242.

BEZDEK, J. C., PAL, S. K. (Eds.), 1992 *Fuzzy Models Pattern Recognition: Methods That Search for Structures in Data*, IEEE.

BISHOP, C. M., 1995, *Neural Networks for Pattern Recognition*. Oxford Press.

BISHOP, C. M., 2006, *Pattern Recognition and Machine Learning*. Springer Science+Business Media.

BLOCH, I., 1999, "On fuzzy distances and their use in image processing under imprecision". *Pattern Recognition*, v. 11, n.32, pp. 1873-1895.

BOHR, H. J.; BOHR, S.; BRUNAK, R.M. *et al.*, 1988, "Protein secondary structure and homology by neural networks". *FEBS Letters*, v.241, n.1, pp. 223-228.

BOVENTI-JR., W. & COSTA, A. H. R., 2000, "Comparação entre métodos de definição de conjuntos nebulosos de cores para classificação de pixels". In: *First Workshop on Artificial Intelligence and Computer Vision*, Atibaia - Brasil.

BRAGA, A. P.; CARVALHO, A. C. P. L. F. & LUDERMIR, T. B., 2000, *Redes Neurais Artificiais – Teoria e Aplicações*. Livros Ténicos e Científicos.

BRANDEN, C. & TOOZE, J., 1999, *Introduction to Protein Structure – 2nd Edition*. Garland Publishing.

BRIDLE, J., 2000, "Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters". In: *Proceedings of the Neural Information Processing Systems*, pp. 211-217. California, USA.

BROOMHEAD, D. S. & LOWE, D., 1988, "Multivariable function interpolation and adaptive networks". *Complex Systems*, v.2, n.1, pp. 321-355.

CAMPBELL, M. K., 2000, *Bioquímica*. Porto Alegre, Artes Médicas Sul.

CASP4, 2000, *Forth Community wide experiment on the critical assessment of techniques for protein structure prediction.* http://predictioncenter.llnl.gov/casp4 .

CASP, 2008, *Community wide experiment on the critical assessment of techniques for protein structure prediction.* http://predictioncenter.llnl.gov . Acessado em Dezembro de 2008.

CHANDONIA, J. M. & KARPLUS, M., 1996, "The importance of larger data sets for protein secondary structure prediction with neural networks". *Protein Science*, v.5, n.1, pp. 768-774.

CHANDONIA, J. M. & KARPLUS, M., 1999, "New methods for accurate prediction of protein secondary structure". *Protein Structure, Function and Genetics*, v.35, n.1, pp. 293-306.

CUFF, J. A. & BARTON, G. J., 2000, "Application of multiple sequence alignment profiles to improve protein secondary structure prediction". *Protein Structure, Function and Genetics*, v.3, n.40 (Aug), pp. 502-511.

CUFF, J. A.; CLAMP, M. E.; SIDDIQUI, A. S.; FINLAY, M. & BARTON, G. J., 1998, "Jpred: A Consensus Secondary Structure Prediction Server", *Bioinformatics* v.14, pp. 892-893.

CYBENCO, G., 1989, "Approximation by superpositions of a sigmoid function". *Mathematics of Control, Signals and Systems*, v.2, n.1, pp. 303-314.

DAYHOFF, M. O.; BARKER, W. C. & HUNT L. T., 1983, "Establishing homologies in protein sequences". *Methods in Enzymology*, v.91, n.1, pp. 524.

DE CAMPOS, T. E, 2001, *Técnicas de Seleção de Características com Aplicações em Reconhecimento de Faces*. Dissertação de Mestrado, Instituto de Matemática e Estatística, Universidade de São Paulo.

DIETMANN, S. & HOLM, L., 2001, "Identification of homology in protein structure classification". *Nature Structural Biology*, v.8, n.1, pp. 953-957.

DUBOIS, D.; PRADE, H. & YAGER, R. R., 1997, *Fuzzy Information Engineering* Wiley Computer Publishing, USA.

DUDA, R. O.; HART, P. E. & STORK, D. G., 2000. *Pattern classification - Second edition*. New York, John Wiley & Sons, Inc.

EIDHAMMER, I.; JONASSEN, I. & TAYLOR, W. R., 2000, "Structure comparison and structure patterns". *Journal of Molecular Biology*, v.7, n.5 (Oct), pp. 685-716.

ESCALIER, V., 1997, *Algorithmes pour la comparaison de structures moléculaires tridimensionnelles*. Thèse de Doctorat, Universitè Paris VII, France

FAHLMAN, S. E., 1988, *An empirical study of learning speed in backpropagation networks*. Technical report, Carnegie Mellow University.

FERREIRA, F. R., 2004, *O uso de rede neural artificial MLP na predição de estruturas secundárias de proteínas*. Dissertação de Mestrado, Departamento de Biofísica da UNESP, São José do Rio Preto.

FERRIS, R. S; CAMPOS, T. E, & CESAR-JR, R.M., 2000, "Detection and tracking of facial features in video sequences". *Lecture Notes in Artificial Intelligence*, v.1973, pp. 129-137.

FISCHER, I.; HENNECKE, F. BANNES, C. *et al*., 2001, *Java Neural Network Simulator – User Manual Version 1.1*. University of Stuttgart, Institute for Parallel and Distributed High Performance Systems.

FISHER, R. A., 1938, "The Statistical utilization of multiple measurements". In: *Annals of Eugenics*, volume 8, pp. 376-386.

FRAENCKEL, A. S., 1993, "Complexity of Protein Folding". *Bulletin of Mathematical Biology*, v.55, n. 6 (Nov), pp. 1199-1210.

FRISHMAN, D. & ARGOS, P., 1997, "Seventy-five percent accuracy in protein secondary structure prediction". *Proteins*, v.27, n. 3 (Mar), pp. 329-335.

FUKUNAGA, K., 1990. *Introduction to Statistical Pattern Recognition*. Academic Press.

GARNIER, J.; OSGUTHORPE, D. J. & ROBSON, B., 1978, "Analysis of the accuracy and implications of simple methods for predicting the secondary structure of globular proteins". *Journal of Molecular Biology*, v.120, n. 1 (Mar), pp. 97-120.

GEMAN, S.; BIENENSTOCK, E. & DOURSAT, R., 1992, "Neural networks and bias-variance dilemma". *Neural Networks*, v.4, n. 1, pp. 1-58.

GOLUB, T. R. *et al.*, 1999, "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring". *Science*, v.286, n. 1, pp. 531-537.

GONZALEZ, R. C. & WOODS, R. E., 1992, *Digital Imaging Processing*. Addison-Wesley Publishing Company.

GRALEWICZ, P., 2004, Quantum computing in neural network. Technical report, http://arxiv.org/PS_cache/quant-ph/pdf/0401/0401127v2.pdf.

GUIMARÃES, K. S.; MELO, J. C. B. & CAVALCANTI, G. D. C., 2003, "Combining few neural nets for effective secondary structure prediction". In: *Proceedings of the Third IEEE Symposium on Bioinformatics and Bioengineering*, pp. 415-420. Maryland, USA.

HAGAN, M. & MANHAJ, M., 1994, "Training feedforward networks with the Marquardt algorithm". *IEEE Transactions on Neural Networks*, v.5, n.6 (Nov), pp. 989-993.

HASSOUN, M. H., 1995, *Fundamentals of artificial neural networks*. Cambridge, MIT Press.

HAYKIN, S., 1999, *Neural networks: a comprehensive foundation, 2nd edition*. Prentice-Hall.

HE, X. & LAPEDES, A., 1991, *Nonlinear modeling and prediction by successive approximations using radial basis functions*. Technical Report, Los Alamos National Laboratory.

HIGGINS, D. & TAYLOR, W., 2000, *Bioinformatics – Sequence, Structure and Databanks*. Oxford University Press.

HOLLEY, L. H. & KARPLUS, M., 1989, "Protein secondary structure prediction with a neural network". *Biophysics*, v.86, n.1 (Jan), pp. 152-156.

HOLLEY, L. H. & KARPLUS, M., 1991, "Neural Networks for Protein Structure Prediction". *Physical Review*, v.2002, n.1, pp. 204-224.

HOLM, L. & SANDER, C., 1993, "Protein structure comparison by alignment of distance matrices". *Journal of Molecular Biology*, v.233, n.1, pp. 123-138.

HUA, S. & SUN, Z., 2001, "A novel method of protein secondary structure prediction with high segment overlap measure: support vector machine approach". *Journal of Molecular Biology*, v.308, n.2 (Apr), pp. 397-407.

JAIN, A. K.; ZONGKER, D. 1997, "Feature selection: evaluation, application and small sample performance". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v.19, n.2, pp. 152-157.

JAIN, A. K.; MURTY, M. N. & FLYNN, P. J., 1999, "Data clustering: a review". *ACM Computing Surveys*, v.31, n.3, pp. 264-323.

JAIN, A. K.; DUIN, R. P. W. & MAO, J., 2000, "Statistical pattern recognition: a review". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v.22, n.1, pp. 4-37.

JAMES, P., 1997, "Protein identification in the post-genome era: the rapid rise of proteomics". *Q Rev Biophys*, v.30, n.4, pp. 279-331.

JONES, D. T., 1999, "Protein secondary structure prediction based on position-specific scoring matrices". *Journal of Molecular Biology*, v.292, n.2 (Sep), pp. 195-202.

KANNAN, S. K. & MYERS, E. W., 1996, "An algorithm for locating nonoverlapping regions of maximum alignment score". *Society for Industrial and Applied Mathematics (SIAM) Journal on Computing*, v.25, n.3 (Jun), pp. 648-662.

KHAN, J. *et al.*, 2001, "Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks". *Nat. Méd.*, v.7, n.1, pp. 673-679.

KHOURI, C. M. B., 2003, *Modelos escondidos de Markov para classificação de proteínas*. Tese de Mestrado, Centro de Informática da UFPE, Recife.

KING, R. & STERNBERG, M., 1996, "Identification and application of the concepts important for accurate and reliable protein secondary structure prediction". *Proteins Science*, v.5, n.11 (Nov), pp. 2298-2310.

KLEYWEGT, G. & JONES, T., 1997, "Detecting folding motifs and similarities in protein structures". *Methods in Enzymology*, v.277, n.11, pp. 525-545.

KONO, H.; DOI, J., 1994, "Energy minimization method using automata network for sequence and side-chain conformation prediction form given backbone geometry". *Proteins*, v.19, n.1, pp. 244-255.

KURTZ, S., 2000, "Computation and visualization of degenerate repeats in complete genomes". In: *Proceedings of the Eight International Conference on Intelligent Systems for Molecular Biology – ISMB2000*, pp. 228-238, San Diego, USA. AAAI Press.

LACROIX, Z. & CRITCHLOW, T., 2003, *Bioinformatics – Managing Scientific Data*. San Francisco, Morgan Kaufmann Publishers.

LANG, S., 1987, *Linear Algebra*. New York, Springer-Verlag.

LEHTONEN, J. V.; DENESSIOUK, K. A.; MAY, A. C. W. *et al.*, 1999, "Finding local structural similarities among families of unrelated protein structures: a generic non-linear algorithm". *Proteins*, v.34, n.3, pp. 341-355.

LESK, A. M., 1995, "Systematic representation of protein folding pattern". *Journal of Molecular Graphics*, v.13, n.1, pp. 159-164.

LOOG, M.; DUIN, R. P. W. & HAEB-UMBACH, R., 2001, "Multiclass linear dimension reduction by weighted pairwise Fisher". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v.23, n.7 (Jul), pp. 762-766.

MARSAN, L. & SAGOT, M. F., 2000, "Algorithms for extracting structured motifs using a suffix tree with application to promoter and regulatory site consensus identification". *Journal of Computational Biology*, v.7, n.1, pp. 345-360.

MCCULLOCH, W. S. & PITTS, W., 1943, "A logical calculus of the ideas immanent in nervous activity". *Bulletin of Mathematical Biophysics*, v.5, n.1, pp. 115-133.

MELO, J. C. B., CAVALCANTI, G. D. C. & GUIMARÃES, K. S., 2003, "Protein secondary structure prediction with ICA feature extraction". In: *Proceedings of the IEEE Workshop on Neural Networks for Signal Processing*, pp. 13-22, Toulouse, France.

MINSKY, M. & PAPERT, S., 1969, *Perceptrons: an introduction to computational geometry*. Massachusetts, MIT Press.

MOOD, J. & DARKEN, C., 1989, "Fast learning in networks of locally-tuned processing units". *Neural Computation*, v.1, n.1, pp. 281-294.

MORAIS, E. C., 2008, "An Optimized Multilayer Perceptron Framework for Protein Secondary Structure Prediction". In: *Posters Session of the BIOMAT 2008*, Campos do Jordão-SP, Brasil.

MORAIS, E. C., 2009, "Statistical Pattern Recognition and MLP Artificial Neural Networks in Protein Secondary Structure Prediction". In: *Posters Session of the BIOMAT 2009*, Brasília-DF, Brasil.

MORAIS, E. C. & MONDAINI, R., 2008, "Optimized Multilayer Perceptrons by Weighted Fisher Criteria for Protein Secondary Structure Prediction". In: *Resumo das Comunicações do CNMAC 2008*, Belém-PA, Brasil.

MORAIS, E. C.; VILELA, S. P. & MONDAINI, R., 2009, "cMLP: Ferramenta de Redes Neurais Artificiais MLP para Predição de Estruturas Secundárias". In: *Anais do CNMAC 2009*, Cuiabá-MT, Brasil.

MORIMOTO, C. H.; YACOOB, Y & DAVIS, L., 1996, "Recognition of head gestures using hidden markov models". In: *ICPR*, Vienna, Austria.

MOUNT, D. W., 2004, *Bioinformatics – Sequence and Genome Analysis*. Second Edition. New York, Cold Spring Harbor Laboratory Press.

MUROGA, S., 1971, *Threshold logic and its applications*. New York, Wiley.

MURZIN, A. G.; BRENNER, S. E.; HUBBARD, T. *et al*, 1995, "SCOP: a structural classification of proteins database for the investigation of sequences and structures". *Journal of Molecular Biology*, v.247, n.1, pp. 536-540.

NELSON, D. L. & COX, M. M., 2000, *Lehninger – Principles of Biochemistry*. *Third Edition*. New York, M. Cox. Worth Publishers.

O'NEIL, M. C. & SONG, L., 2003, "Neural network analysis of lymphoma microarray data: prognosis and diagnosis near-perfect". *BCM Bioinformatics*, v.4, n.1, pp. 13.

PAO, Y., 1989, *Adaptive Pattern recognition and neural networks*. Addison-Wesley.

PEARLMUTTER, B., 1992, "Gradient descent: second order momentum and saturation error". Eds: MOODY, J.E.; HANSON, S. & LIPPMANN, R. *Advances in Neural Information Processing Systems 2*, pp. 887-894. Morgan Kaufmann.

PERLOVSKY, L. I., 1998, "Conundrum of combinatorial complexity". *Trans. on Pattern Analysis and Machine Intelligence* , v.20, n.6, pp. 666-670.

PETERSEN, T. N.; LUNDEGAARD, C.; NIELSEN, M. *et al*, 2000, "Prediction of protein secondary structure at 80% accuracy". *Proteins: Structure, Function and Genetics*, v.41, n.1 (Oct), pp. 17-20.

PETITJEAN, M., 1998, "Interactive maximal common 3D substructure searching with the combined SDM/RMS algorithm". *Journal of Computational Chemistry*, v.22, n.6, pp. 463-465.

POLLASTRI, G.; PRZYBYLSKI, D.; ROST, B. *et al*, 2002, "Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles". *Proteins: Structure, Function and Genetics*, v.47, n.2 (May), pp. 228-235.

PRZYBYLSKI, D.; ROST, B., 2002, "Alignments grow, secondary structure prediction improves". *Proteins: Structure, Function and Genetics*, v.46, n.2 (Feb), pp. 197-205.

QIAN, N. & SEJNOWSKI, T. J., 1988, "Predicting the secondary structure of globular proteins using neural network models". *Journal of Molecular Biology*, v.202, n.4 (Aug), pp. 865-884.

QIAN, N. & SEJNOWSKI, T. J., 1996, "Prediction of Helix in Proteins based on thermodynamic parameters from solution chemistry". *Journal of Molecular Biology*, v.256, n.1, pp. 1157-1168.

RAUDY, S., 1992, "Accuracy of feature selection and extraction in statistical and neural net pattern classification". In: *Proceedings of the Int. Conf. Pattern Recogn.*, pp. 62-70.

RAUDY, S., 1997, "On dimensionality, sample size and classification error of non-parametric linear classification algorithms". *IEEE Transactions on Pattern Anal. Mach. Intell*, v.19, n.1, pp. 667-671.

RAUDY, S. & SKURIKHINA, M., 1992, "The role of the number of training samples on weight initialization of artificial neural net classifier". *RNNS/IEEE Symp. Neuroinform. Neurocomput.*, v.1, n.1, pp. 343-353.

REED, R., 1993, "Pruning algorithms – a survey". *IEEE Transactions on Neural Networks*, v.4, n.5, pp. 740-746.

RIEDMILLER, M. & BRAUN, H., 1993, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm". In: *Proceedings of the IEEE International Conference on Neural Networks*, pp. 586-591, San Francisco, USA.

RIEDMILLER, M., 1994, *Rprop – description and implementation details*. Technical report, University of Karlsruhe.

RIEDMILLER, S. E., 1988, *An empirical study of learning speed in backpropagation networks*. Technical report, Carnegie Mellow University.

RIIS, S. K. & KROGH, A., 1996, "Improving prediction of protein secondary structure using structured neural networks and multiple sequence alignments". *Journal of Molecular Biology*, v.3, n.1 (Spring), pp. 163-183.

RIPLEY, B., 1996, *Pattern recognition and neural networks*. Cambridge University Press.

ROSENBLATT, F., 1958, "The perceptron: a probabilistic model for information storage and organization in the brain". *Psychol. Rev.*, v.65, n.1, pp. 386-408.

ROST, B., 1996, "PHD: predicting one-dimensional protein structure by profile based neural network". *Methods in Enzymology*, v.266, n.1, pp. 525-539.

ROST, B., 1997, “Better 1D predictions by experts with machines”. *Proteins*, v.1, n.1, pp. 192-197.

ROST, B., 1998, “Protein structure prediction in 1D, 2D and 3D 2000”. *The Encyclopedia of Computational Chemistry*, v.3, n.1, pp. 2242-2255.

ROST, B., 2001, “Review: protein secondary structure prediction continues to rise”. *Journal of Structural Biology*, v.134, n.2-3 (Jun), pp. 204-218.

ROST, B. & SANDER, C., 1993, “Improved prediction of protein secondary structure by use of sequence profiles and neural networks”. In: *Proceedings of the National Academy of Sciences of United States of America*, pp. 7558-7562.

ROST, B. & SANDER, C., 1994, “Combining evolutionary information and neural networks to predict protein secondary structure”. *Proteins*, v.19, n.1 (May), pp. 55-72.

ROST, B. & SANDER, C., 2000, “Third generation prediction of secondary structure”. *Protein Structure Prediction: Methods and Protocols*, v.143, n.1, pp. 71-95.

ROST, B. & VA, V. A. E., 2001, “Eva: large-scale analysis of secondary structure prediction”. *Proteins*, v. 5, n.1, pp. 192-199.

ROST, B., YACHDAV, G. & LIU, J. 2004, “The PredictProtein Server”. *Nucleic Acids Research*, v.32 (Web Server Issue), pp. W321-W326.

RUMELHART, D. E.; HINTON, G. E. & WILLIAMS, R. J., 1986, “Learning representations by back-propagation errors”. *Nature*, v.323 n.1, pp. 533-536.

RUMELHART, D. E. & MCCLELLAND, J. L., 1986, *Parallel Distributed Processing - Volume 1: Foundations*. The MIT Press.

SAGOT, M. F., 1998, “Spelling approximate repeated or common motifs using a suffix tree”. In: *Proceedings of the Third Latin American Symposium on Theoretical Informatics*, pp. 111-127, LNCS 1380, Berlin. Spring Verlag.

SALAMOV, A. & SOLOVYEV, V., 1995, "Prediction of protein secondary structure by combining nearest-neighbor algorithms and multiple sequence alignments". *Journal of Structural Biology*, v.247, n.1 (Mar), pp. 11-15.

SANDER, C. & SCHNEIDER, R., 1994, "The HSSP database of protein structure-sequence alignments". *Nucleic Acids Research*, v.22, n.17 (Sep), pp. 3597-3599.

SCHULZ, G. E. & SCHIRMER, R. H., 1979, *Principles of Proteins Structure*. New York, Springer Verlag.

SCOTT, L. B. P., 2003, *Investigação da utilização de algoritmos genéticos e redes neurais artificiais na predição de estruturas protéicas*. Tese de Doutorado, Departamento de Física da UNESP, São José do Rio Preto.

SCOTT, L. P. B.; CHAHINE, J. & RUGGIERO, J., 2007, "Aplicação de redes MLP na predição de estruturas secundárias de proteínas". *Biomatemática*, v.17, n.1 (Aug), pp. 87-100.

SETUBAL, J. C. & MEIDANIS, J., 1997, *Introduction to Computational Molecular Biology*. Boston, PWS Publishing Co.

SIDDIQUI, A. S.; DENGLER, U. & BARTON, G. J., 2001, "3Dee: a database of protein structural domains". *Bioinformatics*, v.17, n.2 (Feb), pp. 200-201.

STEUER, R.; KURTHS, J. *et al.*, 2003, "Observing and interpreting correlations in metabolomic networks". *Bioinformatics*, v.19, n.8, pp. 1019-1026.

STULTZ, C. M.; NAMBUDRIPAD, R.; LATHROP, R. H. & WHITE, J.V., 1997, "Predicting Protein Structure with Probabilistic Models". Chapter in: *Protein Structural Biology in Biomedical Research*. v. 22B, (Editor: E.E. Bittar), JAI Press, Greenwich, pp. 447-506.

TAYLOR, W. & ORENGO, C. A., 1989, "Protein structure alignment". *Journal of Molecular Biology*, v.8, n.1, pp. xx-xx.

THEODORIDIS, S. & KOUTROUMBAS, K., 1999, *Pattern Recognition*. Academic Press, USA.

THOMPSON, J. D.; HIGGINS, D. G. & GIBSON, T. J., 1994, "ClustalW: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position specific gap penalties and weight matrix choice". *Nucleic Acids Research*, v.22, n.22 (Nov), pp. 4673-4680.

UKKONEN, E., 1995, "Online construction of suffix trees". *Algorithmica*, v.14, n.3, pp. 249-260.

VAN'T VEER, L. J. *et al.* 2002, "Gene expression profiling predicts clinical outcome of breast cancer". *Nature*, v.415, n.1, pp. 530-536.

VASCONCELOS, A. T., 2001, "Bioinformática: análise de banco de dados genéticos". *II Escola de Verão: Métodos Computacionais em Biologia*, pp. 47-55.

WANG, J. T. L.; MA, Q.; SHASHA, D. *et al.* 2001, "New techniques for extracting features from protein sequences". *IBM Systems Journal*, v.40, n.2, pp. 426-441.

WANG, Z.; WANG, Y; XUAN, J. *et al.*, 2004, "Optimizing multilayer perceptrons by discriminatory component analysis". In: *Proceedings of the IEEE Workshop on Machine Learning for Signal Processing*, pp. 273-282.

WANG, Z.; WANG, Y; XUAN, J. *et al.*, 2006, "Optimized multilayer perceptrons for molecular classification and diagnosis using genomic data". *Bioinformatics*, v.22, n.6, pp. 755-761.

WEI, J. S. *et al.*, 2004, "Prediction of clinical outcome using gene expression profiling and artificial neural networks for patients with neuroblastoma". *Cancer Res.*, v.64, n.1, pp. 6883-6891.

WIDROW, B. & HOFF, M. E., 1960, *Adaptive switching circuits*. Institute of Radio Engineers, Western Electronic Show and Convention.

XUAN, J.; DONG, Y; KHAN, J. *et al.*, 2004, "Robust feature selection by weight Fisher criterion for multiclass prediction in gene expression profiling". In: *Proceedings of the International Conference on Pattern Recognition*, pp. 291-294.

ZELL, A., 2005, *Stuttgart neural Network simulator*. http://www-ra.informatik.unituebingen.de/SNNS. Acessado em Dezembro de 2005.