UNIVERSIDADE DO VALE DO RIO DOS SINOS

CIÊNCIAS EXATAS E TECNOLÓGICAS

PROGRAMA INTERDISCIPLINAR DE PÓS-GRADUAÇÃO

EM COMPUTAÇÃO APLICADA – PIPCA

SCHEILA DE AVILA E SILVA

Redes Neurais Artificiais aplicadas na caracterização e predição de regiões promotoras

São Leopoldo

2006

UNIVERSIDADE DO VALE DO RIO DOS SINOS

CIÊNCIAS EXATAS E TECNOLÓGICAS

PROGRAMA INTERDISCIPLINAR DE PÓS-GRADUAÇÃO

EM COMPUTAÇÃO APLICADA – PIPCA

SCHEILA DE AVILA E SILVA

Redes Neurais Artificiais aplicadas na caracterização e predição de regiões promotoras

Dissertação apresentada à Universidade do Vale do Rio dos Sinos (UNISINOS) como requisito parcial para obtenção do título de Mestre em Computação Aplicada

Orientador: Prof. Dr. Adelmo Cechin

São Leopoldo

2006

A João Carlos Sartor,

pelo companheirismo e apoio.

AGRADECIMENTOS

Ao meu orientador Prof. Dr. Adelmo Luis Cechin pelo apoio ao longo da realização do trabalho de dissertação.

Ao Prof. Dr. Ney Lemke, pela orientação no primeiro ano de mestrado e pelas relevantes contribuições dadas no trabalho de dissertação realizado durante o segundo ano, não só de nível acadêmico, mas também humano.

À HP Brazil R&D, pela bolsa de mestrado concedida e pelo apoio financeiro para participação em eventos.

Aos colegas de laboratório pelo clima descontraído de trabalho e pelo auxílio em trabalhos das disciplinas realizadas no primeiro ano de mestrado.

Ao Prof. Dr. Günther J. L. Gerhardt, pela orientação durante a bolsa de Iniciação Científica na Graduação e pelo encaminhamento ao curso de mestrado; além de contribuições ao trabalho de dissertação realizado durante o segundo ano, não só de nível acadêmico, mas também humano.

À minha família, pelo apoio em mais esta etapa de minha vida.

À secretária do PIPCA, Sandra Rodrigues, pela cordialidade e eficiência para tratar de questões burocráticas.

RESUMO

A região promotora é uma seqüência de DNA que localiza-se anteriormente a uma determinada região gênica. Ela é responsável pelo início do processo de transcrição de um gene ou conjunto de genes. Assim, ela também atua como um elemento regulador da expressão gênica. O estudo da regulação da expressão gênica é relevante porque é essencial para a compreensão da maquinária vital dos seres vivos, já que a diferença entre duas espécies está mais relacionada em como e quando seus genes estão “ativos” ou “inativos” do que com a estrutura destes em si. Embora exista métodos computacionais para a predição de genes com boa acurácia, o mesmo não é conseguido para os promotores. Esta dificuldade deve-se ao pequeno e pouco conservado padrão das seqüências, gerando assim resultados com alto número de falsos positivos. Além dos motivos consensuais, os promotores possuem características físicas que os diferem de seqüências não-promotoras. No entanto, estas ainda não são amplamente utilizadas no problema de predição *in silico* de promotores. Este trabalho tem como objetivo a utilização de Redes Neurais Artificiais para predizer e caracterizar promotores de *Escherichia coli* através de duas abordagens: uma utilizando a codificação ortogonal e outra utilizando os valores de estabilidade da seqüência promotora. Para a caracterização, realiza-se a extração de regras do tipo *if...then*. Os resultados obtidos neste trabalho são de 80% para a predição com codificação ortogonal e 74% para a predição com a estabilidade. As regras extraídas, para as duas abordagens, atestam que mesmo degenerados, os motivos consensuais são determinantes para que a rede classifique uma determinada seqüência como promotora.

Palavras-chave: promotores procarióticos, rede neural artificial, predição

ABSTRACT

The promoter region is localized few base pairs before the gene region. It is responsible by initiate the gene expression process, thus, it plays a regulatory role. The study about the gene expression regulation is a great area, since it can assist in the comprehension of complex metabolic network presented by several organisms and, because the difference between two different species is how and when your genes are turn off and turn on than your structure. The computational methods to gene prediction have a good accuracy, but this is not achieved in the promoter prediction. This difficulty occurs because the length of promoter and the degenerate pattern presented, thus the results presented a great number of false positives. This work aims employed Neural Networks to promoter prediction and recognition of *Escherichia coli* by two approaches: whit the orthogonal codification and stability values of the promoter sequence. For characterization, realize the extraction rules of type if … then. The results in this work are: 80% and 74% for prediction whit orthogonal codification and stability, respectlly. The rules extracted for two approaches are accordingly whit the biologic knowledge. Despite the short and incompletely conservation, the consensual motifs are important to classify a given sequence like promoter.

Key words: promoters prokaryotic, neural network, prediction

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

A	Nucleotídeo Adenina
AM	Aprendizado de Máquina
BP	Algoritmo *Back-propagation*
C	Nucleotídeo Citosina
DNA	Ácido desoxirribonucléico
E. *coli*	*Escherichia coli*
FN	Falsos Negativos
FP	Falsos Positivos
G	Nucleotídeo Guanina
HMM	Modelos Ocultos de Markov
k-FCV	k-*fold-cross-validation*
MLP	*Multilayer Perceptron*
pb	Pares de base
RMS	Erro Médio Quadrático
RN	Rede Neural
RNA	Ácido ribonucléico
RNAm	Ácido ribonucléico mensageiro
RNAp	Enzima RNA polimerase
ROC	*Receiver Operating Characteristic*
RP_c	Complexo fechado do promotor
RP_o	Complexo aberto do promotor
SVM	Máquinas de Vetor de Suporte *(Suport Vector Machine)*
T	Nucleotídeo Timina
TLS	Sítio de início da tradução
TSS	Sítio de início de transcrição
VN	Verdadeiros Negativos
VP	Verdadeiros Positivos

SUMÁRIO

1 INTRODUÇÃO

Os recentes avanços na Genética e na Biologia Molecular disponibilizam uma grande quantidade de dados biológicos. No entanto, não tornam a descoberta do conhecimento (confirmação ou rejeição de hipóteses científicas ou proposição de novas hipóteses) uma questão fácil, já que os fenômenos biológicos são muito complexos e requerem a integração de muitas áreas do conhecimento (BARRERA et al, 2004).

A interface interdisciplinar mais antiga entre a Biologia e as Ciências Exatas e, talvez, a mais conhecida, é a Bioestatística. Gradualmente nos últimos anos, a Biologia tem utilizado, com avidez, as ferramentas proporcionadas pela Informática e pela Matemática para a resolução de problemas nos mais diversos campos: desde a Genética até a Ecologia. Como as metodologias tradicionais em Biologia Molecular não permitem a análise total dos dados, seja pelo alto custo ou pelo tempo de análise, as ferramentas de Aprendizado de Máquina (AM) ganham aplicabilidade ao problema, já que são capazes de aprender de forma automatizada a partir de dados disponíveis e levantar hipóteses relevantes sobre os mecanismos biológicos ainda ocultos (BARRERA et al, 2004).

A molécula de DNA (ácido desoxirribonucléico) não contém apenas genes (pedaços de DNA que decodificam informação), mas também elementos que

governam essa decodificação, os chamados elementos regulatórios. O conjunto destes elementos é conhecido como regulação da expressão gênica (LEWIN, 2001). O estudo da expressão gênica, bem como de sua regulação, é um ponto importante de estudo, já que a transcrição (primeiro passo da expressão gênica) é regulada pela interação entre seqüências específicas da molécula de DNA e proteínas que aí se ligam. A mais conhecida destas seqüências é a região promotora (SIVARAMAN et al, 2005). De uma maneira simplificada, pode-se dizer que estas localizam-se anteriormente à região gênica e interagem com a enzima RNA polimerase (RNAp), iniciando o processo de decodificação do gene (DE ROBERTIS, 1993).

Como os promotores estão intimamente ligados à expressão gênica, eles podem atuar também, como elementos regulatórios. Fazendo uma analogia, os elementos *downstream* (como os genes) representam a memória de um computador e os elementos *upstream* (como os promotores) seriam os programas que atuam nesta memória. O estudo dos promotores então, se faz necessário para prover modelos sobre a constituição do "programa" e de como este opera (HOWARD e BENSON, 2003). A região promotora possui locais específicos e relativamente conservados, que auxiliam no reconhecimento e na ligação da RNAp nesta região. Além destes locais, os promotores possuem algumas características estruturais próprias, que os diferem de regiões não-promotoras (LEWIN, 2001; KANHERE e BANSAL, 2005a).

As mais variadas abordagens computacionais têm sido empregadas para reconhecer estas regiões e, predizer se uma região é ou não promotora. Dentre estas técnicas, pode-se destacar Análise Probabilística, Reconhecimento de Padrões e AM. Embora haja progressos na predição e análise de promotores, estas ainda estão longe de possuir uma alta acurácia. A predição de elementos

regulatórios (como os promotores) pode auxiliar na compreensão do complexo processo de regulação da expressão gênica, bem como melhorar o reconhecimento de regiões codificadoras, ou seja, os genes (BURDEN et al, 2005).

Esta dissertação tem como tema a aplicação de Redes Neurais Artificiais (RNs) na predição e caracterização de seqüências promotoras de organismos procariotos, como a bactéria *Escherichia coli* (E. *coli*). Além da predição das seqüências promotoras, este trabalho extrai regras a partir das RNs treinadas. Assim, pode-se compreender o que foi aprendido no processo de aprendizagem pela rede e verificar quais elementos da seqüência possuem um papel determinante no reconhecimento de uma dada seqüência como promotora. Isto auxilia o pesquisador na inferência de questões relevantes sobre os promotores e na validação do conhecimento biológico sobre estes.

Nas próximas seções, descrevem-se os objetivos e a motivação para a realização deste trabalho de dissertação.

1.1 OBJETIVO GERAL

Reconhecer e caracterizar regiões promotoras através da composição de nucleotídeos da seqüência promotora codificados ortogonalmente e dos valores de estabilidade utilizando a abordagem de Redes Neurais Artificiais e, após a extração de regras para a identificação do conhecimento aprendido pela RN.

1.2 OBJETIVOS ESPECÍFICOS

- Preparação dos dados de entrada para a realização do treinamento*;*

- Determinação da melhor arquitetura de RNs para o reconhecimento de regiões promotoras de *E.coli;*

- Extração de regras para compreensão dos mecanismos utilizados para o reconhecimento de promotores com a utilização do ambiente R e do *software* WEKA;

- Emprego de informações de estabilidade da seqüência promotora como parâmetros de entrada para treinamento da RN;

1.3 MOTIVAÇÃO

Um dos maiores desafios da era pós-genômica é a determinação de quando, onde e como os genes são “ligados” e “desligados”. A diferença entre duas espécies está muito mais relacionada com a transcrição de seus genes do que com a estrutura destes em si. Assim, o estudo da regulação gênica contribui para a construção do conhecimento a respeito da funcionalidade dos genes em diferentes espécies, na questão da diferenciação celular em organismos multicelulares, na resposta celular frente às mudanças ambientais, entre outras questões (HOWARD e BENSON, 2003; COTIK et al, 2005; LEWIN, 2001).

Dentre os elementos regulatórios da transcrição, os promotores ganham destaque, porque são os maiores responsáveis pelo início de todos os processos de transcrição e da expressão correta dos genes (LEWIN, 2001). Portanto, as ferramentas de AM ganham aplicabilidade ao problema, já que são capazes de aprender de forma automatizada a partir de dados disponíveis e levantar hipóteses sobre mecanismos biológicos ainda ocultos.

Nas abordagens aplicadas na análise de promotores procarióticos, apesar de serem conhecidas, as informações físicas da molécula de DNA como a diferença de estabilidade, curvatura, deformabilidade e composição de dinucleotídeos entre as regiões *upstream* e *downstream* não são consideradas. Portanto, a investigação de metodologias que considerem as informações estruturais e funcionais sabidamente características de promotores, e a extração de regras de inferência para propor hipóteses biológicas referentes ao problema de reconhecimento e caracterização de regiões promotoras, é relevante e de caráter promissor, conforme o estado da arte.

1.4 ORGANIZAÇÃO DO TEXTO

O texto está organizado como descrito a seguir:

O capítulo 2 apresenta alguns conceitos fundamentais sobre Biologia Molecular relacionados com a expressão gênica em organismos procariotos. Neste capítulo, dá-se enfoque na caracterização estrutural dos promotores procarióticos e sua relação com a expressão gênica. O capítulo seguinte traz a revisão bibliográfica das principais abordagens *in silico* para a análise de promotores de *E. coli*. Neste capítulo, destaca-se a metodologia de RNs, a qual será fundamentada no capítulo 4. O capítulo 4 apresenta conceitos fundamentais sobre as RNs: princípios, arquitetura e algoritmos, além de conceitos sobre a extração de regras. O capítulo 5 descreve o protocolo realizado para utilizar a RN como ferramenta reconhecedora de seqüências promotoras e os resultados obtidos são analisados e discutidos no capítulo 6. Para encerrar, as conclusões finais são reunidas no capítulo 7.

2 A EXPRESSÃO GÊNICA EM ORGANISMOS PROCARIOTOS

Em 1953, James Watson e Francis Crick, baseados em vários trabalhos da época sobre o DNA, publicaram na revista científica *Nature* um trabalho denominado *"Molecular Structure of Nucleic Acids - A Structure for Desoxyribose Nucleic Acid*" (Estrutura molecular dos ácidos nucléicos - Uma Estrutura para o Ácido Desoxirribonucléico), que descrevia pela primeira vez a estrutura do DNA (DE ROBERTIS, 1993).

O DNA ou ácido desoxirribonucléico é uma molécula que existe praticamente em todas as células dos organismos vivos e de alguns vírus, com a função de armazenar a informação genética (DE ROBERTIS, 1993). As seções seguintes descrevem a molécula de DNA e o processo de transcrição que ocorre em organismos procarióticos.

2.1 A MOLÉCULA DE DNA

O DNA é composto por unidades repetitivas chamadas nucleotídeos. Estas unidades são formadas por um grupo fosfato, um açúcar e uma base nitrogenada, sendo esta última que diferencia os nucleotídeos uns dos outros. As bases nitrogenadas podem ser de quatro tipos: adenina (A), guanina (G), citosina (C) e timina (T), conforme a figura 2.1.

As bases do DNA são classificadas como purinas ou pirimidinas, sendo a diferença entre elas a presença de um ou dois anéis heterocíclicos. As pirimidinas (C e T) possuem apenas um anel, enquanto as purinas (A e G) possuem dois anéis fusionados. Os nucleotídeos são ligados formando fitas, sendo a molécula composta por duas fitas complementares e antiparalelas (dupla-hélice). A Figura 2.1 ilustra a organização da molécula de DNA.

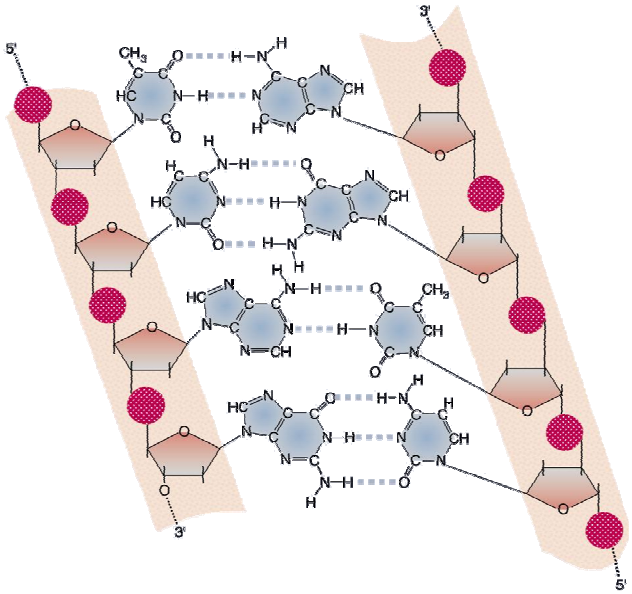

Figura 2.1 - Esquema representando a molécula de DNA. Nela, a seqüência representada é T-C-A-G na fita da esquerda (5' - 3') e sua seqüência complementar A-G-T-C na fita da direita (3' - 5') Aqui, o grupo fosfato é representado por um círculo vermelho, o açúcar é representado pelo pentagrama e as bases nitrogenadas estão no centro da molécula. As pontes de hidrogênio estão representadas por pontilhados (LEWIN, 2001).

Uma das características do DNA é que o nucleotídeo "A" sempre estabelece ligação com o nucleotídeo "T" e o nucleotídeo "C" sempre estabelece ligação com "G". Essa ligação é feita através de pontes de hidrogênio, sendo a ligação "A – T" constituída de duas pontes de hidrogênio e a ligação "C – G" formada por três pontes de hidrogênio (ver Figura 2.1). A quantidade de pontes implica na estabilidade da molécula, sendo que as ligações com três pontes de hidrogênio são

mais fortes do que as que possuem duas. Mesmo que as pontes de hidrogênio sejam fracas, o grande número delas existente no DNA é suficiente para manter as duas cadeias unidas.

Outra característica da molécula é a sua polaridade, que é evidenciada no grupo fosfato – molécula carregada negativamente. Diz-se que o DNA é uma molécula polar porque uma das fitas possui o grupo fosfato e o grupo hidroxila (também com carga elétrica negativa) na direção 5' (cinco linha) ⟶ 3' (três linha) do açúcar. A fita complementar possui estes grupos na direção 3' (três linha) ⟶ 5' (cinco linha). Estas informações também podem ser vistas na Figura 2.1.

Algumas porções do DNA são fundamentais para a produção de determinadas unidades biomoleculares específicas. Estes fragmentos de DNA são denominados genes. São os genes que determinam o molde para a seqüência de aminoácidos dos seus produtos. Na maioria das vezes, estes produtos são proteínas que realizam uma função específica na célula: estrutural, regulatória ou catalítica.

A expressão dos genes necessários para uma determinada função celular é denominada transcrição e varia de acordo com o estágio de desenvolvimento celular ou com condições ambientais. O controle de qual gene deve ser expresso em um determinado momento compreende um conjunto de mecanismos que torna este processo complexo até mesmo para organismos unicelulares, como as bactérias. Este processo é conhecido como regulação da expressão gênica.

A próxima seção descreve o processo de transcrição dos genes em organismos procariotos e o papel do promotor para o seu desencadeamento.

2.2 TRANSCRIÇÃO DOS GENES

Quando um gene é expresso, sua informação é copiada na forma de ácido ribonucléico (RNA) que por sua vez, dirige a síntese dos produtos elementares dos genes. Em Biologia Molecular, o termo **transcrição** é utilizado como sinônimo de síntese de RNA mensageiro (RNAm) e **tradução** é sinônimo de síntese de proteína, que ocorre no ribossomo celular. Este processo de transcrição-tradução é denominado como dogma central da Biologia Celular, que pode ser visualizado na Figura 2.2.

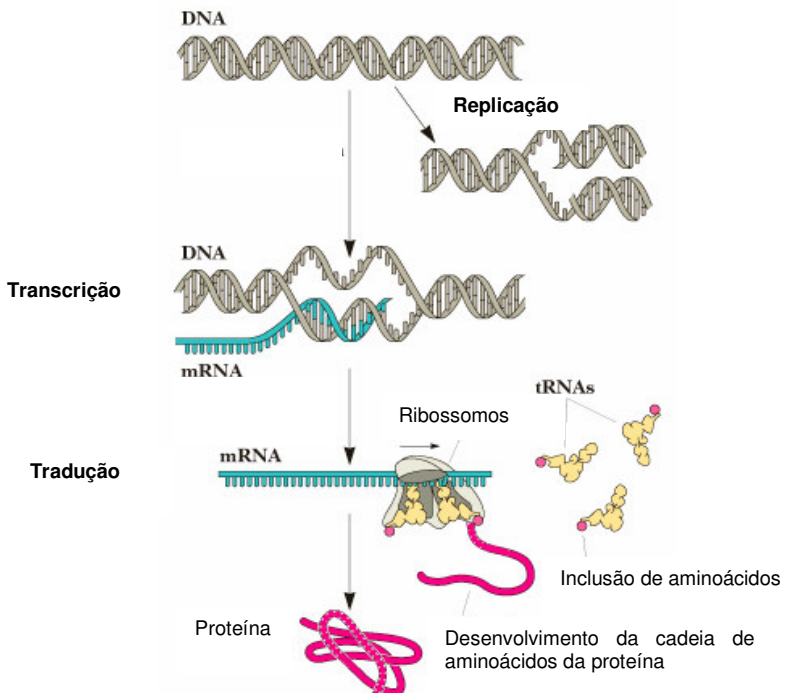

Figura 2.2 – Dogma central da biologia molecular. O DNA é capaz de se auto-replicar, de produzir um RNAm a partir de um fragmento de DNA (gene), ou seja, realizar o processo de transcrição – e este RNAm decodifica uma cadeia de aminoácidos que formará uma proteína que desempenhará alguma função celular, esta última etapa chamada de tradução (LEWIN, 2001).

A iniciação da transcrição dos genes inicia quando a enzima RNAp liga-se em seqüências específicas do DNA, denominadas de promotores. Em *E. coli*, a ligação

da RNAp ocorre dentro de uma região que se estende desde cerca de 70 pares de base (pb) antes do sítio de início da transcrição (TSS) até cerca de 30 pb além dele. Por convenção, os pb de DNA que correspondem ao início de uma molécula de RNAm recebem números positivos, sendo esta parte do DNA denominada de região *downstream.* Já a região *upstream* corresponde aos nucleotídeos que precedem o sítio de início da transcrição recebem números negativos (ver Figura 2.3). A região promotora, portanto, estende-se entre as posições -70 e +30.

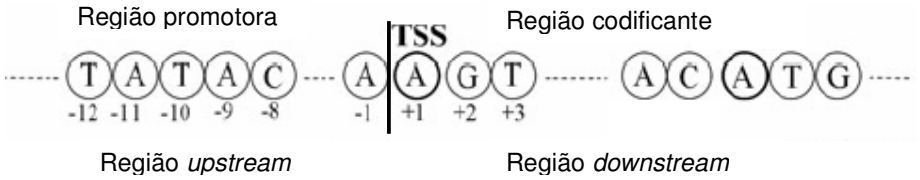

Figura 2.3 – Representação da região promotora para uma única fita de DNA, em *E. coli.* Aqui estão as regiões *upstream* (antecedente ao sítio de início da transcrição) e *downstream* (adjacente ao sítio de início da transcrição) e a numeração que cada nucleotídeo recebe. (BURDEN et al, 2005- modificado).

A enzima RNAp desempenha um importante papel no início da transcrição como reconhecedora das seqüências promotoras. Ela contém cinco unidades básicas: duas subunidades α, uma subunidade β, uma β' e uma subunidade ω. Essas cinco subunidades formam o *core* da RNAp. Além destas, há um grupo designado como fator σ, que liga-se transitoriamente ao *core* e direciona a enzima para sítios de ligação específicos do DNA. Quando o fator σ está associado à RNAp, ela passa a ser chamada de RNAp holoenzima. Na Tabela 2.1, estão as subunidades da RNAp, seu gene codificante e sua função no processo de transcrição, e na Figura 2.4 está um esquema desta enzima.

Tabela 2.1 – Descrição das subunidades da RNA polimerase holoenzima de *E. coli*. (LEHNINGER et al, 2000).

Subunidade	Gene codificante	Função na RNAp
α	rpoA	Ligação dos elementos regulatórios
β	rpoB	Ligação dos nucleotídeos – formação das ligações fosfodiéster
β'	rpoC	Ligação à fita molde
σ	rpoD	Reconhecimento da região promotora e iniciação da transcrição
ω	rpoZ	Acréscimo na força de associação entre as subunidades

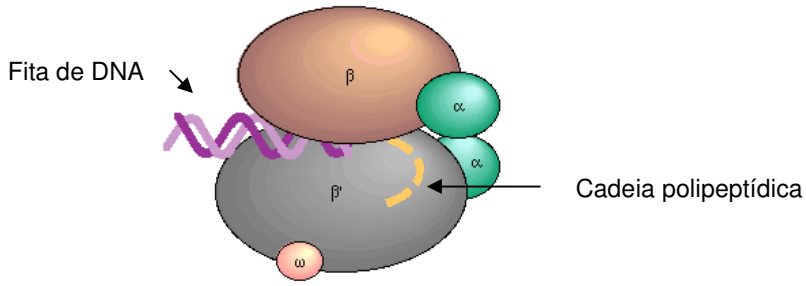

Figura 2.4 – Esquema da RNAp de organismos procariotos como *E. coli*. Aqui mostra-se as 5 subunidades componentes da enzima (KEGG, 2006).

Existem diferentes tipos de fatores σ (ver Tabela 2.2) que podem se ligar com o *core* da RNAp, sendo cada um associado a uma classe de promotores que regulam a expressão de um determinado conjunto de genes necessários em um dado momento celular.

Tabela 2.2 – Fatores σ de *E. coli*. (LEHNINGER et al, 2000).

Fator σ	Gene codificante	Momento celular de ativação
28	rpoF	Resposta a estresse por falta de alimento
32	rpoH	Resposta a estresse por choque térmico
38	rpoS	Genes dos flagelos
54	rpoN	Assimilação de nitrogênio
70	rpoD	Sigma constitutivo

A transcrição possui dois momentos principais, cada um com múltiplas etapas. O início do processo ocorre quando a RNAp holoenzima se liga ao promotor, formando um complexo fechado do promotor – RPc – no qual o DNA ligado está intacto. Após, há a formação do complexo aberto – RP_o – onde o DNA desta região está parcialmente desenrolado. Em seguida, inicia-se a transcrição deste complexo e após a inserção dos dois primeiros nucleotídeos na molécula de RNA transcrita, ocorre a liberação do fator σ. O término da transcrição ocorre quando é encontrada uma seqüência de nucleotídeos denominada de região terminadora. Este processo está ilustrado na Figura 2.5.

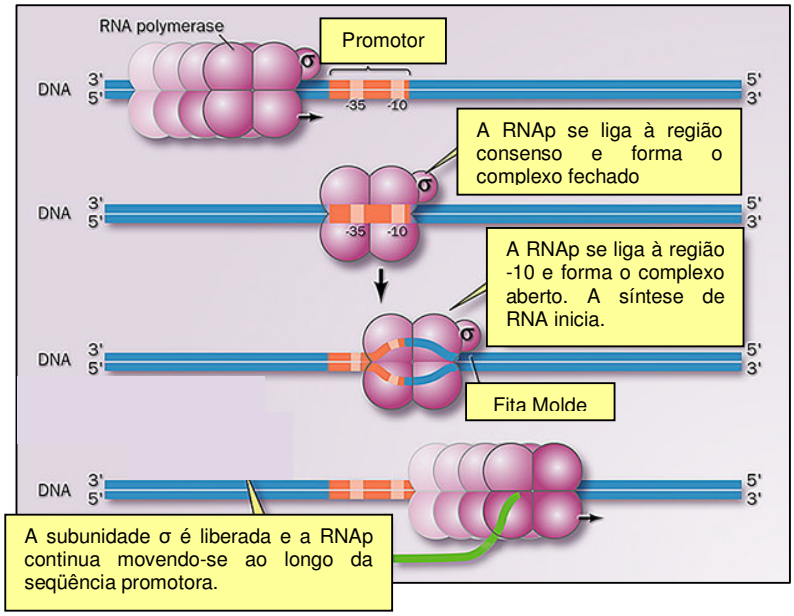

Figura 2.5 – Etapas da iniciação da transcrição pela RNAp de *E. coli*. (LEHNINGER et al, 2000).

Neste trabalho, o interesse está nas características da região promotora. Assim, a seguir são apresentados os conhecimentos moleculares sobre a estrutura e função dos promotores.

2.3 PROMOTORES PROCARIÓTICOS

Promotores são regiões do DNA que antecedem os genes, sendo reconhecidos por proteínas específicas, como a RNAp. Ela possui uma especificidade com determinadas regiões do promotor devido à ligação do fator σ. Um promotor procariótico típico é constituído de 3 regiões características: uma seqüência de 6 nucleotídeos (hexâmero) centrada em –35 do ponto inicial de transcrição (+1), outro hexâmero centrado em –10 e a seqüência que separa os hexâmeros (espaçador), conforme ilustrado na Figura 2.6.

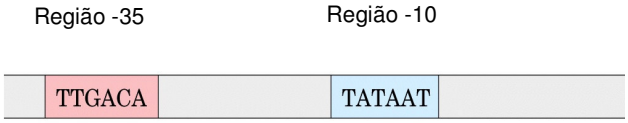

Figura 2.6 – Região promotora procariótica e seus elementos principais (LEHNINGHER et al, 2000).

Análises e comparações das seqüências da classe mais comum de promotores bacterianos (aqueles reconhecidos pela RNAp holoenzima contendo σ^{70}) revelam semelhanças nos dois hexâmeros citados anteriormente. Embora as seqüências não sejam idênticas para todos os promotores bacterianos, certos nucleotídeos particularmente comuns em determinadas posições formam uma seqüência consenso (ver Figura 2.7). O modelo biológico padrão para estes promotores é a seqüência ***TTGACA*** para a região -35, ***TATAAT*** para a região -10 e um espaçamento entre estes hexâmeros de 17 nucleotídeos.

Muitas linhas independentes de pesquisa atestam a importância funcional das seqüências -35 e -10 (KANHERE e BANSAL, 2005a; KALATE et. al, 2003; LEHNINGER et al, 2000). O hexâmero -35 funciona como sinal para reconhecimento pela RNAp e o hexâmero -10 permite converter o complexo fechado em complexo

aberto. Além disso, a distância entre eles parece ser relevante, apesar do tamanho variável e da falta de conservação (LEWIN, 2001).

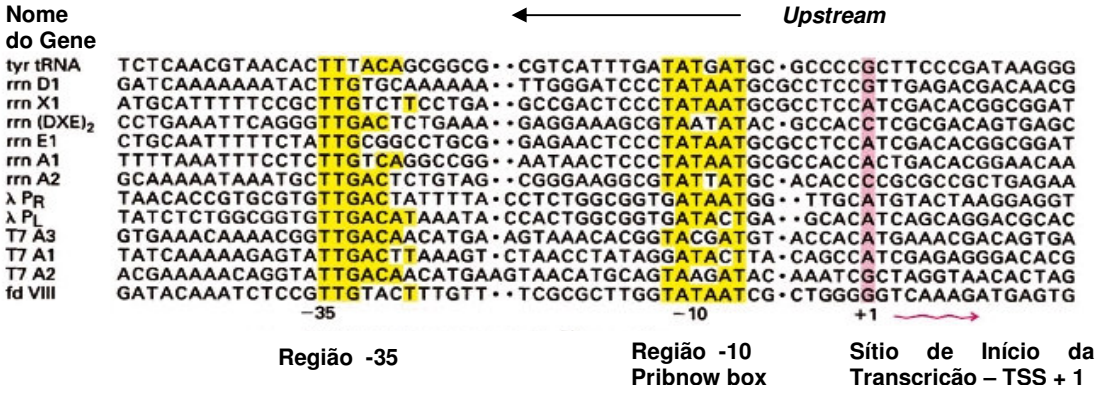

Figura 2.7 – Promotores típicos de *E. coli* reconhecidos pela RNAp holoenzima σ70. As seqüências consenso -10 e -35 são mostradas para alguns promotores. As regiões espaçadoras mostram-se com tamanho variado. O primeiro nucleotídeo que codifica o RNAm (+1) está destacado.

Variações na seqüência consenso podem afetar a eficiência da ligação da RNAp. Uma mudança em apenas um pb pode diminuir a velocidade de iniciação em várias ordens de grandeza. A seqüência do promotor, desta forma, estabelece um nível basal de expressão dos genes, sendo considerado um promotor forte aquele mais perto da seqüência consensual e um promotor fraco aquele que possui mais de três nucleotídeos diferentes do consenso (LEWIN, 2001).

Apesar da simplicidade com a qual os livros de Biologia Molecular (LEWIN, 2001; LEHNINGER et al, 2000; DE ROBERTIS, 1993) descrevem os promotores, percebe-se em trabalhos experimentais como os de Borukhov e Lee (2005), Burgess e Anthony (2001), Murakami et al (2002), Naryshkin et al (2000), que a interação entre a enzima e a seqüência promotora é um processo complexo, que envolve a interação de vários locais da seqüência com a RNAp.

Além disso, a seqüência constituinte não é a única característica determinante da região promotora. Existem diferenças estruturais entre as regiões *upstream* (promotora) e *downstream* (não-promotora) que podem ser consideradas para melhorar a predição das seqüências promotoras e caracterizá-las. É muito difícil acreditar que apenas os motivos consensuais sejam os responsáveis pela interação RNAp-promotor, já que estes motivos são pequenos e não completamente conservados. É possível que as seqüências vizinhas a estes motivos também estejam envolvidas neste processo de interação RNAp-promotor. Algumas destas propriedades que podem ser consideradas são: a deformabilidade, estabilidade e curvatura da região *upstream* (KANHERE e BANSAL,2005a).

Trabalhos como os de Kanhere e Bansal (2005a, 2005b) e JÁUREGUI et al (2003) mostram que as regiões promotoras são menos estáveis que as regiões gênicas. Quanto à curvatura da molécula, os fragmentos de DNA vizinhos à TSS mostram-se mais curvados que os demais fragmentos e, em relação à deformabilidade, as regiões promotoras são menos maleáveis que as codificantes. A importância destas propriedades para os promotores e para o processo de transcrição, está descrito a seguir.

Um importante passo durante a transcrição é a formação do RP_o, que envolve a separação das fitas de DNA. Esta separação é um processo termodinamicamente desfavorável e ocorre sem nenhuma ajuda energética de fonte externa. Aqui, a pouca estabilidade da seqüência promotora pode auxiliar no início de separação das fitas. Outra propriedade, como a curvatura, pode ser definida como a dupla fita curvada em um axis helicoidal. Muitas seqüências, de organismos eucariotos e procariotos mostram que as regiões *upstream* são mais curvadas que as regiões gênicas. Já a deformabilidade, refere-se ao afrouxamento com o qual a molécula

pode realizar uma curva em alguma direção. Sabe-se que a deformabilidade é importante para a ligação de fatores de transcrição e evidências experimentais sugerem que a seqüência promotora se enrola ao redor da RNAp (KANHERE e BANSAL, 2005a, 2005b).

Apesar destas evidências em relação à estrutura da seqüência promotora, estas informações não são amplamente utilizadas na predição e reconhecimento dos promotores, conforme descrito no próximo capítulo, que apresenta os principais trabalhos referentes à análise de promotores procariotos *in silico*.

3 ANÁLISE DE PROMOTORES *IN SILICO*: REVISÃO BIBLIOGRÁFICA

O estudo de promotores é um dos aspectos fundamentais para a compreensão da expressão gênica, já que um dos passos para a predição *ab initio* dos genes é desenvolver melhor os métodos de predição de promotores e da TSS (KANHERE e BANSAL, 2005b). Ainda que os promotores sejam de importância indiscutível, a habilidade em identificá-los é menos desenvolvida que a de encontrar regiões codificantes. A maior dificuldade no seu reconhecimento *in silico* é que sua seqüência é muito curta e não apresenta-se completamente conservada. Portanto, há uma alta probabilidade de encontrar seqüências similares em outras regiões genômicas que não as promotoras. Alguns problemas inerentes à predição de promotores ocorrem porque o processo de transcrição é iniciado por interações específicas entre a RNAp e a seqüência de DNA na região promotora, já que estas interações são complexas e seus elementos componentes altamente degenerados (KANHERE e BANSAL, 2005b; HOWARD e BENSON, 2003; SIVARAMAN et al, 2005; BURDEN, et al, 2005).

Os algoritmos para predição de promotores, que baseiam-se apenas na similaridade de pb da seqüência apresentam baixa especificidade e/ou sensibilidade, além de um alto índice de falsos positivos (BURDEN et al, 2005). Portanto, a investigação de metodologias que consideram informações estruturais e funcionais

das regiões promotoras, torna-se relevante para o entendimento da complexa maquinária biológica envolvida no reconhecimento destas seqüências.

A literatura apresenta muitas abordagens para o reconhecimento e predição de promotores. Dentre estas pode-se citar: *(i)* metodologia baseada em sinal, que opera no reconhecimento de sinais relativamente conservados através de alinhamento e homologia entre promotores previamente identificados; *(ii)* AM, que usa conjunto de informações estruturais e funcionais disponíveis sobre as seqüências promotoras para "aprender" a reconhecê-los automaticamente e produzir hipóteses relevantes sobre estas seqüências. Aqui, encontram-se as metodologias de Redes Neurais (RN), Modelo Oculto de Markov (HMM), *Suport Vector Machine* (SVM), e metodologias híbridas.

A seguir, estão apresentados os principais métodos computacionais encontrados na literatura de predição de promotores - abordagem baseada em sinal e AM - acompanhados da discussão das limitações que tornam este campo de pesquisa ainda latente. Uma das poucas referências bibliográficas com a utilização de informações sobre a estabilidade da seqüência promotora para o seu reconhecimento também será apresentado (KANHERE e BANSAL, 2005b). Como a metodologia utilizada foi desenvolvida pelos próprios autores e não se enquadra em nenhum dos dois métodos acima mencionados, este será discutido em uma seção separada dos demais.

3.1 RECONHECIMENTO BASEADO EM SINAL

A metodologia de reconhecimento de promotores baseado em sinal emprega principalmente a comparação do conteúdo de diferentes seqüências promotoras. Os trabalhos clássicos e alguns mais recentes estão apresentados a seguir.

3.1.1 Seqüência Consenso

A determinação da seqüência consenso é o método clássico para analisar os promotores. Conforme Lisser e Margalit (1993), este método consiste em alinhar um conjunto de seqüências identificadas previamente como promotora e, posteriormente, pesquisar por regiões conservadas em seu conteúdo. Cada coluna no alinhamento fornece a variação encontrada nesta posição do promotor.

Apesar dos hexâmeros -35 e -10 terem sido encontrados facilmente, esta é uma técnica muito simples e imprecisa (OPPON, 2000). Lisser e Margalit (1993) apresentam a distribuição de bases para os consensos dos hexâmeros, de acordo com 298 promotores identificados experimentalmente em sua pesquisa. Os resultados estão sumarizados na Tabela 3.1.

Tabela 3.1 – Freqüências em porcentagem (%) para os hexâmeros -35 e -10, de acordo com Lisser e Margalit (1993), totalizando 298 promotores identificados experimentalmente para *E.coli*. Os nucleotídeos da linha horizontal representam a seqüência consenso e os da primeira coluna da esquerda representam as substituições que podem ocorrer nos promotores.

	Região -35						**Região -10**					
	T	**T**	**G**	**A**	**C**	**A**	**T**	**A**	**T**	**A**	**A**	**T**
A	10	6	9	56	21	54	5	76	15	61	56	6
C	10	7	12	17	54	13	10	6	11	13	20	7
G	10	8	61	11	9	16	8	6	14	14	8	5
T	70	79	18	16	16	16	17	12	60	12	15	82

A importância e o pioneirismo deste trabalho são indiscutíveis para a análise dos promotores, já que existem trabalhos atuais com outras metodologias, que comprovam a existência e importância destes motivos, como os trabalhos de Cotik et al (2005); Eskin et al (2003) e Sivaraman et al (2005). Mas somente a análise da seqüência para a descoberta de novos promotores é uma abordagem limitada, por que: *(i)* a variação dos nucleotídeos é grande; *(ii)* assume a independência entre bases adjacentes; *(iii)* não permite a presença de múltiplos elementos dos promotores, inserções, deleções ou espaço variável entre os elementos e *(iv)* o resultado pode variar de acordo com o método de alinhamento.

3.1.2 Matriz de Posições Ponderadas

A metodologia de matriz de posições ponderadas é baseada na metodologia da seqüência consenso, mas com algumas modificações. Conforme o trabalho de Hertz e Stormo (1999), as matrizes de posições ponderadas assumem que cada linha da matriz corresponde a um dos quatro nucleotídeos e cada coluna a um alinhamento. Os elementos da matriz são os pesos utilizados para pontuar uma seqüência teste, conforme uma medida que quantifica a aderência ao modelo.

A pontuação é dada pela soma dos pesos de cada letra alinhada em cada posição, conforme pode ser visto na Figura 3.1 apresentada a seguir.

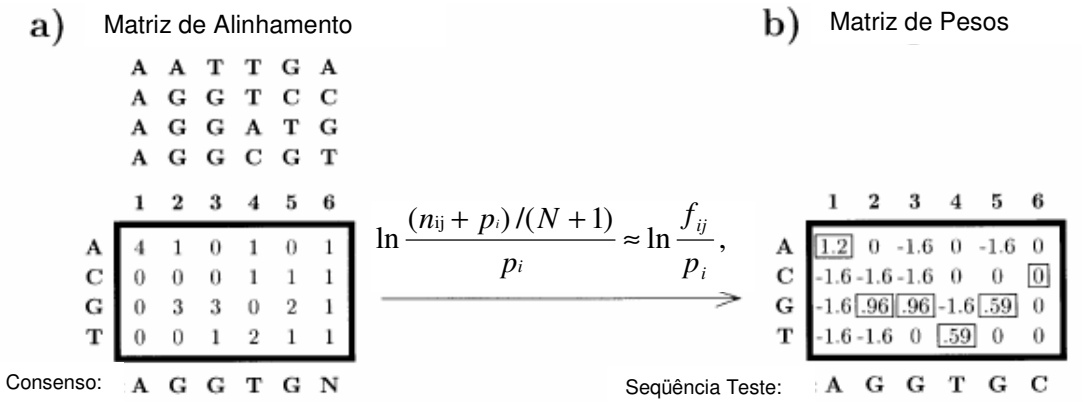

Figura 3.1 – Exemplo da transformação da matriz de alinhamento para a matriz de posições ponderadas para a seqüência teste AGGTGC. Em (a) está uma tabela de alinhamento para os 4 hexâmeros localizados no topo da figura. Embaixo da matriz está a seqüência consenso correspondente ao alinhamento (*N* indica que não há nucleotídeo preferencial). (b) A matriz de pesos derivada da matriz de alinhamento (a). A fórmula para transformar a matriz de alinhamento em uma matriz de pesos está sobre a linha e representada na equação 3.1. Os números destacados são os responsáveis pela pontuação global da seqüência. (Hertz e Stormo, 1999).

A transformação ilustrada na Figura 3.1, acima, foi criada a partir da fórmula:

$$\ln \frac{(n_{ij} + p_i)/(N+1)}{p_i} \approx \ln \frac{f_{ij}}{p_i}, \qquad \textbf{(3.1)}$$

onde n_{ij} é o número de vezes que a letra *i* é encontrada na posição *j* do alinhamento; p_i é a probabilidade *a priori* da letra *i*, no caso 0,25 para cada base do DNA, *N* é o número total de seqüências analisadas e $f_{ij} = n_{ij} \div N$ é a freqüência da letra *i* na posição *j*.

Uma versão modificada desta metodologia foi empregada em Xing *et al* (2005). Neste trabalho, foi utilizado o alinhamento local (LAPP) para estabelecer seqüências consensuais e predizer promotores para Baculovirus (vírus de insetos). Além da matriz de pesos, a identidade no alinhamento recebeu valor +5 e o não-pareamento, valor -4. Dada esta pontuação, os pares de segmento máximo (MSP) são definidos como a pontuação mais alta de identidade de pares. Outra estratégia

utilizada pelos autores é o cálculo da pontuação da predição do promotor (PPS), sendo este definido como:

$$PPS = \frac{\text{soma da pontuação MSP}}{\text{Tamanho do promotor predito (pb)}}. \quad \textbf{(3.2)}$$

Com esta metodologia, foi obtido índice de acerto de 66% para as seqüências promotoras e um número de falsos positivos de aproximadamente 50%.

As limitações da predição de promotores por matriz de posições ponderadas são quase as mesmas que as citadas para análise da seqüência consenso, já que considera apenas o alinhamento de seqüências.

3.2 ANÁLISE POR APRENDIZADO DE MÁQUINA

Aqui, descreve-se as metodologias de AM mais empregadas atualmente na predição de promotores.

3.2.1 Modelos Ocultos de Markov – HMMs

Um Modelo de Markov descreve um processo que pode ser executado em um número de estados num tempo determinado e é dado por $M = (Q, \_, \pi, a, b)$ (HAYKIN, 1999).

Onde:

- $Q = \{q_0,..., q_n\}$ é um conjunto de n estados;

- $\_ = \{\sigma_1,..., \sigma_m\}$ é o alfabeto finito de saída;

- (π_i) são as probabilidades iniciais dos estados;

- (a_{ij}) são as probabilidades de transição do estado *i* para cada estado *j* possível, dado que o modelo está em *i*;

- (b_{ik}) são as probabilidades de emissão dos elementos *k* pertencentes ao alfabeto no estado *i*.

O modelo encontra-se inicialmente no estado q_0, e ao passar pelos demais, forma uma seqüência de caracteres emitidos em cada um dos estados, composta de elementos de _. Como observa-se apenas esse conjunto de símbolos gerados e não a seqüência de estados, esse tipo de modelo de Markov é dito *escondido*. A Figura 3.2 mostra a estrutura de um HMM.

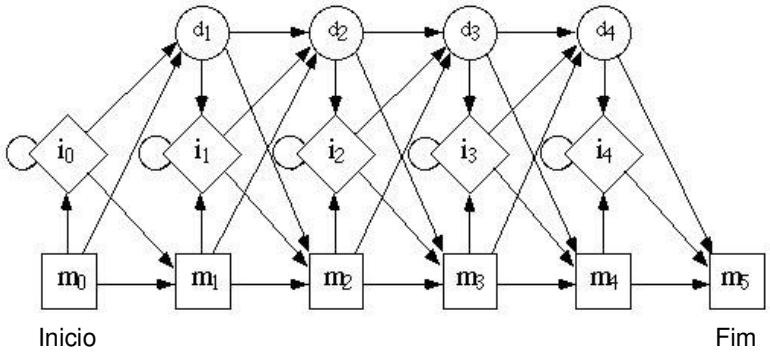

Figura 3.2 – Estrutura do HMM para seqüências biológicas. Além da arquitetura, são apresentados os três tipos de estados: pareamento (m_0 até m_5), inserção (i_0 até i_5) e deleção (d_0 até d_5).

Trabalhos utilizando a metodologia de HMM mostram que esta consegue caracterizar seqüências desconhecidas com base no grupo de treinamento (PEDERSEN et al, 1996). Na questão de exatidão, o trabalho de Reis e Lemke (2004) mostra que é possível obter uma exatidão média de 80%na predição.

A vantagem da metodologia de HMM é que esta apresenta uma capacidade de capturar regularidades em seqüências de caracteres, considerando a variação nos símbolos observados em cada estado. No entanto, uma das restrições do HMM

é o tamanho do conjunto de treinamento, visto a grande quantidade de parâmetros que precisam ser estimados (REIS, 2005). Além disso, há uma dificuldade na incorporação de outras características dos promotores no algoritmo do HMM, como a composição dos dinucleotídeos ou trinucleotídeos da seqüência e informações sobre a estabilidade.

3.2.2 Máquinas de suporte vetorial – (*Suport vector machine)*

O algoritmo das Máquinas de Vetor de Suporte (SVM) foi proposto por Boser, Guyon e Vapnik em 1992 e pode ser utilizado para classificações de padrões e regressão linear. Basicamente, as SVM são uma máquina linear com algumas propriedades muito interessantes. No caso das classificações, a idéia principal é construir um hiperplano como superfície de decisão, de tal forma que a margem de separação entre exemplos positivos e negativos seja máxima. As SVMs podem fornecer um bom desempenho de generalização em problemas de classificação de padrões, apesar de não incorporarem conhecimento do domínio do problema (HAYKIN, 1999).

Gordon et al (2003) usaram uma SVM com núcleo de uma função de alinhamento. Neste trabalho, foram tomados dois conjuntos de dados: *(i)* promotores e regiões codificadoras e *(ii)* promotores e regiões intergênicas. A metodologia empregada por eles mostra uma média de erro de 16,5% e de 18,6%, respectivamente aos conjuntos de dados usados.

Kiryu et al (2005) utilizou SVMs para verificar a existência de alguma correlação entre o grau de conservação da seqüência e a expressão do respectivo gene. Ao contrário do que trazem os livros de Biologia Molecular (LEWIN, 2001;

LEHNINGER et al, 2000; DE ROBERTIS, 1993) estes autores não encontraram nenhuma correlação entre a seqüência promotora e o nível de expressão gênica.

3.2.3 Metodologia Híbrida

Uma metodologia pode ser considerada híbrida, se incorpora várias abordagens de AM, buscando aperfeiçoar os resultados. A escolha das abordagens deve considerar que as deficiências apresentadas por uma metodologia sejam minimizadas pela(s) outra(s). Cotik et al (2005) usaram uma metodologia híbrida para análise das regiões promotoras de organismos procariotos. Neste trabalho, eles propõem a ferramenta HPAM, que combina RN, Lógica Fuzzy e Computação Evolucionária para descobrir motivos representativos dos promotores. A justificativa para a escolha destas metodologias baseia-se na eficiência das RNs em representarem padrões imprecisos e incompletos, na flexibilidade e interpretabilidade dos modelos *fuzzy* e na capacidade da computação evolucionária em identificar exemplos ótimos de um modelo, pela busca de acordo com critérios múltiplos. A análise dos promotores procarióticos revelou, não surpreendentemente, os hexâmeros -10 e -35 como característica destas seqüências.

3.2.4 Redes Neurais

As RNs são um sistema de AM inspirado no funcionamento de redes neurais biológicas. É um modelo computacional paralelo constituído de elementos interconectados, chamados de neurônios. As conexões, entre eles, possuem pesos que são ajustados na etapa de treinamento. Pode-se afirmar que as RNs aprendem a partir dos exemplos e apresentam alguma capacidade de generalização do conjunto de treinamento (WU e MCLARTY, 2000).

As primeiras aplicações de RNs na predição de promotores, como apresentados nos trabalhos de Demeler e Zhou (1991) e O'Neill (1991), apesar da arquitetura simples dos primeiros algoritmos, obtiveram uma alta acurácia na predição, mas o número de falsos positivos foi igualmente alto. Outras abordagens de RN foram apresentadas por Mahadevan e Ghosh (1994) que usaram uma combinação de duas RNs para a identificação de promotores de *E. coli.* Todos os promotores deste trabalho tinham espaçamento entre 15-21 nucleotídeos entre os hexâmeros característicos. A primeira RN predizia os hexâmeros consensuais, enquanto a segunda foi designada para o reconhecimento da seqüência inteira (65 nucleotídeos), sendo o espaço entre os hexâmeros variável. Uma vez usada a informação da seqüência inteira, ocorreu dependências entre as bases em várias posições. Isto refletiu em um treinamento pobre e uma predição realizada por duas redes sem neurônios na camada oculta.

Pedersen e Engelbrecht (1995) predisseram o local de início da transcrição (TSS), a medida do conteúdo da informação e identificaram novos sinais característicos correlacionados com o local de início. Para isso, foram usados dois diferentes esquemas de codificação, um com janelas 1 até 51 nucleotídeos e outro com uma janela de 65 nucleotídeos. Uma idéia interessante, neste trabalho, foi a medida do conteúdo de informação relativa dos dados de entrada, pelo uso da habilidade da RN para aprender corretamente, como avaliado pelo coeficiente de correlação do teste máximo.

Outra ferramenta baseada em RNs é o *Neural Networks Promoter Prediction* (NNPP). Oppon (2000) executou um teste neste sistema a partir do conjunto composto de 31 seqüências de 75 bases, sendo 5 regiões promotoras e 26 regiões codificantes de *E.coli*. Com um limiar de corte de 6,0 o NNPP acerta classificar uma

seqüência como promotora 60% das vezes e em afirmar que não é promotora em 50% das vezes. Uma provável explicação para este baixo desempenho, além do baixo número de amostras, é a especificidade do sistema, que foi desenvolvido especificamente para um organismo. Em 2005, Burden et al melhorou este sistema incorporando nele a informação sobre a distância entre o sítio de início de transcrição (TSS) e o sítio de início da tradução – TLS – (primeiro nucleotídeo do gene). Com um conjunto de dados de 771 promotores, eles melhoraram a predição em 60%, quando comparado ao trabalho de Oppon (2000), e reduziram o número de falsos positivos.

A vantagem das RNs é que elas podem aprender a reconhecer padrões degenerados, imprecisos e incompletos, os quais são característicos dos promotores. Além disso, permitem rápido desempenho em grandes seqüências genômicas (COTIK et al*,* 2005). Como desvantagem, pode-se citar a necessidade de sincronismo nos dados de entrada. Como os padrões dos promotores não estão sempre em lugares fixos (os hexâmeros possuem um espaçador de tamanho variável), esta falta de sincronismo na posição dos hexâmeros consensuais dificulta a aprendizagem da RN. Outros fatores dizem respeito à determinação do número de camadas ocultas e do número de interações, já que não existe uma metodologia formal para estas definições, cabendo ao usuário a determinação destes parâmetros heuristicamente.

3.3 METODOLOGIA UTILIZANDO A INFORMAÇÃO DE ESTABILIDADE

Kanhere e Bansal (2005b) desenvolveram uma metodologia baseada nas diferenças de estabilidade entre as regiões promotoras e gênicas. Eles calcularam a energia livre (estabilidade) entre duas regiões do genoma de um organismo,

conforme as equações (3.3), (3.4), (3.4) e (3.6). Os resultados obtidos por eles mostram que a estabilidade é uma medida melhor que os motivos conservados para diferenciar regiões promotoras e não-promotoras.

$$D(n) = E1(n) - E2(n). \qquad \textbf{(3.3)}$$

onde,

$$E1(n) = \frac{\sum_{n}^{n+49} \Delta G^{o}}{50} \qquad \textbf{(3.4)}$$

$$E2(n) = \frac{\sum_{n+99}^{n+119} \Delta G^{o}}{100} \qquad \textbf{(3.5)}$$

onde,

$$\Delta G^{0} = \Delta G^{0}{}_{ini} + \Delta G^{0}{}_{sym} + \sum_{i=1}^{n-1} \Delta G^{0}{}_{ij+1} \qquad \textbf{(3.6)}$$

onde,

$\Delta G^{0}{}_{ini}$ é a iniciação da energia livre para os dinucleotídeos e recebe valor +1.96 kcal/mol.

$\Delta G^{0}{}_{sym}$ recebe valor +0.43 kcal/mol e é aplicado se a fita-dupla é auto-complementar.

$\Delta G^{0}{}_{ij}$ é a variação de energia livre para os dinucleotídeos de tipo *ij*.

n é o nucleotídeo da seqüência promotora.

A equação que mostra a fórmula de como obter os valores de ΔG^{o} é descrita novamente no capítulo 5 (equação 5.1), pois faz parte da metodologia deste trabalho de dissertação. A metodologia desenvolvida pelos autores para a análise e predição de promotores está esquematizada na Figura 3.3, a seguir.

Quando comparada às outras metodologias descritas neste capítulo, eles conseguem um aumento significativo na predição de promotores. O resultado da predição chega a 90% de acerto. No entanto, esta metodologia somente se aplica em grandes seqüências, já que cada seqüência analisada por eles tinha 1000 pb (do nucleotídeo -500 até o nucleotídeo +500). Caso a seqüência promotora possuísse menos de 500 pb antes ou após a TSS ela era excluída do experimento. Assim, pensamos que a metodologia deve se enquadrar aos dados disponíveis e não enquadrar alguns dados na metodologia para obter a maior exatidão.

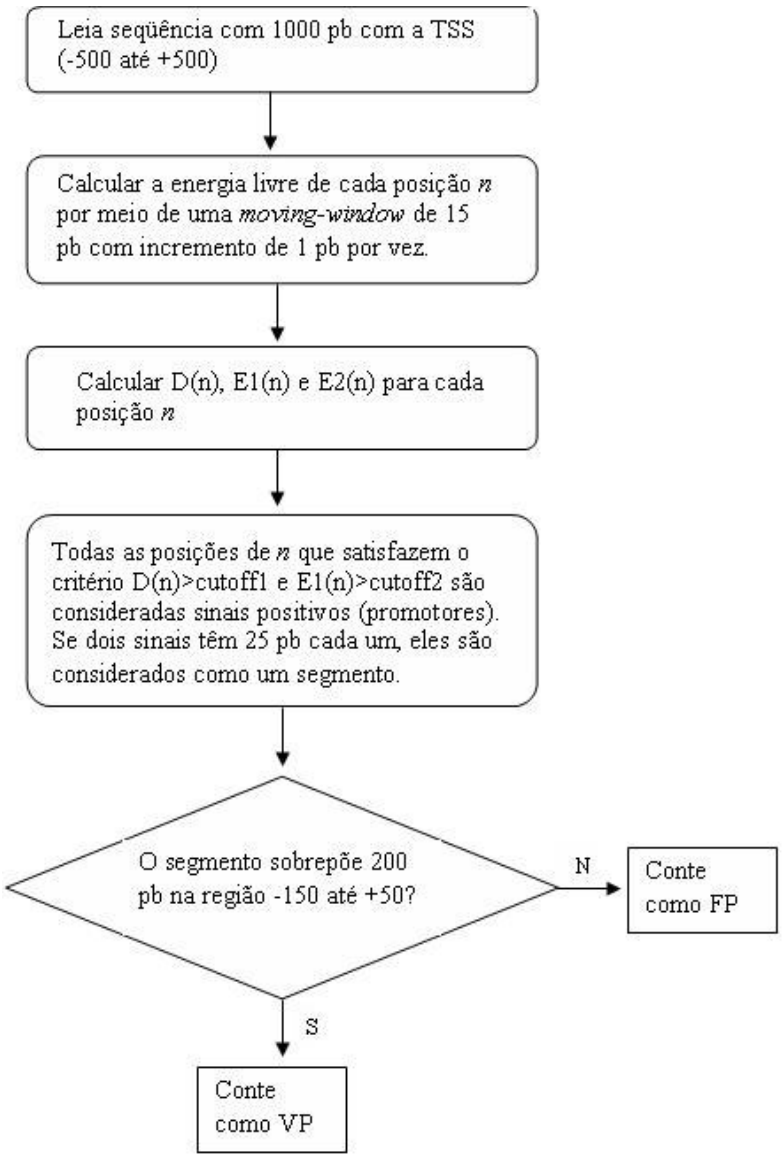

Figura 3.3 – Fluxograma que ilustra a metodologia desenvolvida pelos autores Kanhere e Bansal (2005b) utilizando a informação de estabilidade da seqüência promotora.

3.4 CONSIDERAÇÕES ADICIONAIS

A revisão bibliográfica sobre os métodos computacionais de análise de promotores procarióticos mostra uma diversidade de técnicas empregadas. Percebe-se, em todos os resultados, a presença dos hexâmeros -35 e -10 como elementos conservados entre os promotores. Muitos pesquisadores, como Oppon (2000), mostram que um modo de aumentar a eficiência no treinamento dos modelos para

identificar promotores – quando se tem um conjunto de treinamento mínimo – é integrar diferentes sistemas de predição, como Redes Neurais (RN), HMMs, análise estatística e, além disso, incorporar outras características físicas dos promotores.

Neste capítulo, não há descrição de trabalhos sobre a extração de regras a partir de RNs treinadas para a predição de promotores porque não foi encontrado nenhum artigo relacionado até o término da revisão bibliográfica.

Este trabalho pretende incorporar características físico-químicas, como a estabilidade e também extrair regras de inferência das RNs treinadas para compreender o processo de classificação. Para a concretização dos objetivos, concluímos que a metodologia de RNs mostra-se mais apropriada. A partir destas observações, no próximo capítulo, descreve-se alguns conceitos básicos do uso das RNs.

4 REDES NEURAIS ARTIFICIAIS: FUNDAMENTOS E APLICAÇÕES

As Redes Neurais Artificiais foram originalmente desenvolvidas com o objetivo de modelar o processamento de informação e aprendizagem do cérebro. Trata-se de um modelo computacional aplicável a uma ampla variedade de áreas, como Engenharia, Economia e Biologia. Nesta última, principalmente em problemas de análise de seqüências e reconhecimento de padrões. Nas demais áreas, por exemplo, as RNs podem ser aplicadas na síntese e reconhecimento de fala, interface adaptativa entre humanos e sistemas físicos complexos, aproximação de funções, entre outros (BALDI e BRUNAK, 2001).

Este capítulo apresenta conceitos fundamentais sobre as RNs aplicadas na predição e reconhecimento de promotores. O estudo em questão apresenta conceitos que auxiliam na compreensão da metodologia empregada e possibilita uma melhor discussão da mesma. Além da definição de RN, brevemente introduzida no capítulo 3, apresentam-se conceitos relacionados com: neurônios, arquiteturas e sobre o processo de aprendizado que ocorre em dois tipos de arquiteturas de rede: os *Perceptrons* e as redes multicamadas, incluindo o algoritmo *back-propagation*.

4.1 DEFINIÇÃO

Uma RN é um processador paralelamente distribuído constituído de unidades simples, que tem a propensão natural de armazenar conhecimento experimental e torná-lo disponível para o uso (HAYKIN, 1999). Em um nível mais elementar, uma RN consiste de redes com unidades interconectadas envolvidas no tempo. A conexão de uma unidade *i* com uma unidade *j* usualmente começa com um peso sináptico (ou somente peso) denotado por W_{ji}. Assim, pode-se representar uma RN como um grafo direcionado com peso ou arquitetura (MOUNT 2000).

4.2 ARQUITETURA DAS REDES NEURAIS

As RNs consistem de grupos ou camadas (*layers)* de unidades de processamento com (ou algumas vezes sem) conexões entre os grupos. A unidade básica de uma camada é um neurônio artificial. Estas unidades, como os neurônios reais, têm conexões de entrada (dendritos) e conexões de saída (axônios). Também como neurônios reais, as unidades da rede neural também têm alguma forma de processamento interno, que cria um sinal de saída como uma função do sinal de entrada. Entretanto, há duas diferenças fundamentais entre neurônios reais e uma unidade da rede neural: *(i)* a saída de um neurônio biológico é um pulso de sinal modulado (série de pulsos de amplitude fixa) com sua freqüência mudando em resposta aos sinais recebidos pelos dendritos, enquanto que o neurônio artificial tem como saída um número; *(ii)* a saída de um neurônio biológico muda continuamente com o tempo e já a de um artificial apresenta mudanças somente em um intervalo discreto de tempo, conforme é ilustrado na Figura 4.1. (WU e MCLARTY, 2000).

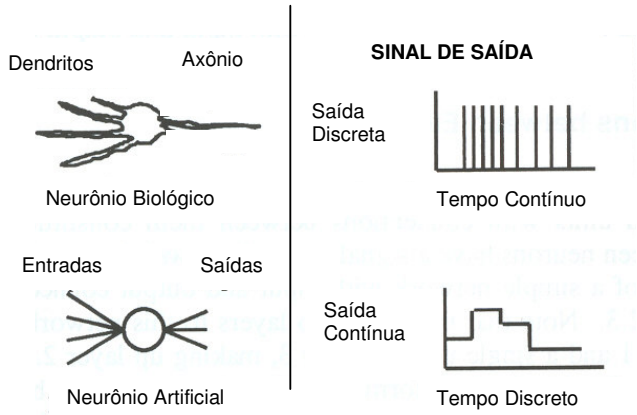

Figura 4.1 – Analogia entre neurônios biológicos e artificiais. Aqui compara-se a estrutura e o sinal de saída (WU e MCLARTY, 2000).

Uma RN é caracterizada pelo *(i)* padrão de conexões entre os neurônios (chamado de arquitetura), *(ii)* método de determinação de pesos nas conexões (chamado de treinamento ou aprendizagem) e *(iii)* sua função de ativação. Esses parâmetros estão descritos ao longo desta e das próximas seções.

Os neurônios (ver ilustração 4.2) são conectados por **vínculos** orientados. Um vínculo da unidade *j* para a unidade *i* serve para propagar a **ativação** a_j desde *j* até *i*. Cada vínculo também tem um peso numérico W_{ji} associado a ele, o qual determina a intensidade e o sinal da conexão. Especificamente, um sinal a_j na entrada da sinapse *i* conectada ao neurônio *j* é multiplicada pelo peso sináptico W_{ji} (RUSSEL, 2003 e HAYKIN, 1999). Após, cada unidade *i* calcula inicialmente uma soma ponderada de suas entradas:

$$in_i = \sum_{j=0}^{n} W_{ji} a_j \,. \qquad \textbf{(4.1)}$$

Então ela aplica uma função de ativação *g* a essa soma para derivar a saída:

$$a_i = g(in_i) = g\left(\sum_{j=0}^{n} W_{ji} a_j \right). \qquad \textbf{(4.2)}$$

É importante ressaltar que há a inclusão de parâmetro externo do neurônio artificial, um ***bias*** W_{0i} (ver Figura 4.2) conectado a uma entrada fixa a_0= -1. O termo W_{0i} define o limite real para a unidade, no sentido de que a unidade é ativada quando a soma ponderada de entradas "reais" $\sum_{j=1}^{n} W_{ji} a_j$ excede W_{0i}.

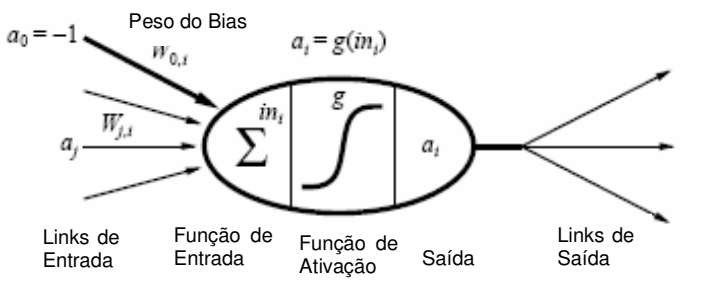

Figura 4.2 – Modelo de um neurônio artificial. A ativação da saída da unidade é $a_i = g(\sum_{j=0}^{n} W_{ji} a_j)$, onde a_j é a ativação de saída da unidade *j* e W_{ji} é o peso no vínculo da unidade *j* até essa unidade. (RUSSELL, 2003).

A função de ativação *g* é projetada para atender duas aspirações: primeiro, a unidade de estar "ativa" (próxima de +1) quando as entradas positivas forem recebidas e negativas (próxima a 0) quando as entradas "erradas" forem recebidas. Em segundo lugar, a ativação precisa ser não-linear, caso contrário a RN inteira entrará em colapso, tornando-se uma função linear simples (RUSSEL, 2003 e HAYKIN, 1999). Conforme descrito em Haykin (1999), existem três tipos básicos de funções de ativação, que são detalhados a seguir e podem ser visualizadas nas figuras 4.3 até 4.5. A função de ativação, representada por *g*, define a saída de um neurônio em termos do campo local induzido (in_i), ou seja, entrada dos neurônios.

1) *Função de Limiar:* Conforme a figura 4.3, para este tipo de função de ativação, tem-se:

$$g(in_i) = \begin{cases} 1 \text{ se } in \geq 0 \\ 0 \text{ se } in < 0 \end{cases} \quad \textbf{(4.3)}$$

Esta função assume apenas valores 0 ou 1.

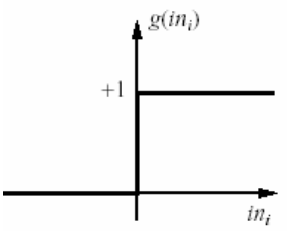

Figura 4.3 – Função de limiar (RUSSELL, 2003).

2) *Função Linear por partes:* para este tipo de função de ativação, tem-se:

$$g(in_i) = \begin{cases} 1, & in \geq +1/2 \\ in, + \dfrac{1}{2} > & in > -1/2 \\ 0, & in \leq -1/2 \end{cases} \quad \textbf{(4.4)}$$

onde assume-se que o fator de amplificação dentro da região linear de operação é a unidade. Esta forma de função de ativação pode ser vista como uma aproximação de um amplificador não-linear. A forma desta função de ativação pode ser vista na Figura 4.4.

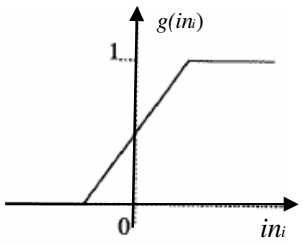

Figura 4.4 – Função linear por partes

3) *Função Sigmóide:* esta é a forma mais comum de função de ativação utilizada na construção de RNs. Ela é definida como uma função estritamente crescente que exibe um balanceamento adequado entre comportamento linear e

não-linear. Um exemplo de função sigmóide é a função logística, conforme ilustrado na Figura 4.5. A equação para esta função é:

$$g(in) = \frac{1}{1 + e^{-in}} \qquad \textbf{(4.5)}$$

Enquanto a função de limiar assume apenas valores 0 e1, a função sigmóide assume valores contínuos entre 0 e 1 e a inclinação final desta função se dá pelos ajustes dos pesos.

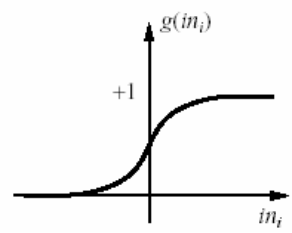

Figura 4.5 – Função Sigmóide (RUSSELL, 2003).

Como visto nesta seção, enquanto a metáfora com o cérebro ainda permanece como uma útil fonte de inspiração é claro hoje que os neurônios artificiais usados nas redes neurais estão longe dos neurônios biológicos. A próxima seção descreve as arquiteturas de *Perceptron* e Redes Neurais Multicamadas.

4.2.1 *Perceptrons* e Redes Neurais Multicamadas

Uma rede com todas as entradas conectadas diretamente às saídas é chamada de *Perceptron* (ver Figura 4.6(a)). Tendo em vista que cada unidade de saída é independente das outras, cada peso afeta apenas uma das saídas. Conforme Haykin (1999), o *Perceptron* foi objeto de intensa pesquisa durante os anos 50 e 60, mas em 1969, Minsky e Papert provaram matematicamente que este tipo de estrutura de processamento apresenta limitações importantes e só pode ser aplicada com sucesso em problemas linearmente separáveis.

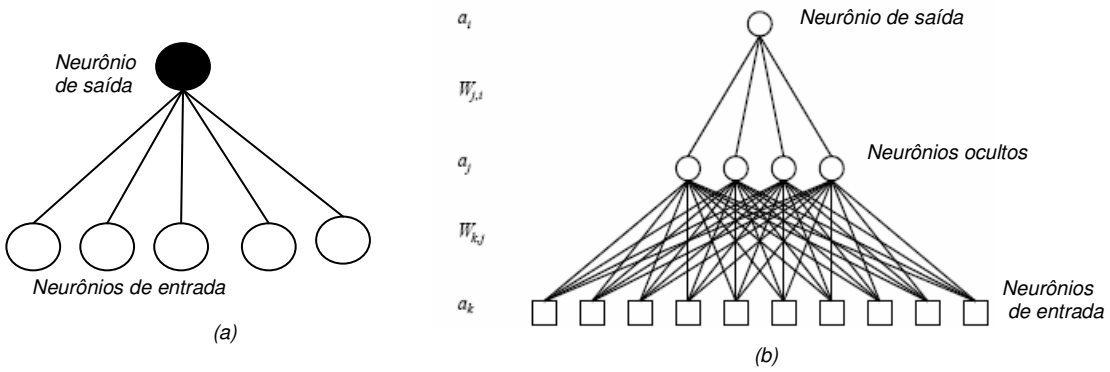

Figura 4.6 – (a) Modelo de rede com duas camadas (*Perceptron) e,* (b) rede com três camadas - *Multilayer Perceptron* (MLP) (RUSSELL, 2003).

Uma arquitetura com melhor capacidade de generalização do que a *Perceptron*, constitui-se de redes com múltiplas camadas, chamadas de *Multilayer Perceptron* (MLP), sendo o caso mais comum aquelas que envolvem uma única camada oculta, conforme ilustrado na Figura 4.6(b). Segundo Hornik (1989), as RNs com uma única camada oculta são aproximadores universais, pois aproximam qualquer função com precisão arbitrária. A função dos neurônios ocultos é intervir entre a entrada externa e a saída de maneira útil. As vantagens de adicionar camadas ocultas é que ela aumenta o espaço de hipóteses que a rede pode representar e, assim, é capaz de extrair estatísticas de ordem elevada. Isto é particularmente valioso quando o tamanho da camada de entrada é grande. Quando se tem uma rede com muitas camadas ocultas, estas são menos eficientes computacionalmente, já que requerem um tempo de computação maior e apresentam menor capacidade de generalização quando comparadas às redes com uma camada oculta. Além disso, a extração das regras da rede se torna mais difícil.

A rede da figura 4.6 (b) é dita totalmente conectada, no sentido de que cada um dos nós de uma camada da rede está conectado a todos os nós da camada

adjacente seguinte. Entretanto, se alguns dos elos de comunicação (conexões sinápticas) estiverem faltando na rede, dizemos que a rede é parcialmente conectada.

Existem diferentes modelos de arquiteturas, como: recorrente, *feed-forward* e *layered*. A recorrente contém *loops* direcionados, e uma desprovida destes *loops* é chamada de *feed-forward.* Uma arquitetura é dita *layered* se as unidades são particionadas em classes, também chamadas camadas, e os padrões de conectividade são definidos entre as classes. Uma arquitetura *feed-forward* não é necessariamente *layered* (MOUNT, 2000).

A próxima seção explica sobre o processo de treinamento das RNs em geral e algumas especificidades das arquiteturas *Perceptron* e MLP. Também mostra o algoritmo *back-propagation* e sua derivação matemática.

4.3 TREINAMENTO DE REDES NEURAIS

Segundo Wu e McLarty (2000), a idéia fundamental por trás do aprendizado ou treinamento, para todas as arquiteturas de RN, é atribuir valores a um conjunto de pesos (inicializado normalmente de forma aleatória), aplicar os dados de entrada à rede e verificar como esta responde a determinados conjuntos de pesos. Se o desempenho não for satisfatório, então os pesos devem ser modificados pelo algoritmo específico da arquitetura e repetir o procedimento. Este procedimento deve ser repetido até que algum critério de parada pré-especificado seja atingido.

A passagem de todos os vetores dos dados de entrada através da rede é chamado de uma *época*. Alterações nos pesos podem ser feitas a cada padrão processado (treinamento *on-line)* ou após uma época inteira (treinamento em lote),

sendo esta última o procedimento mais utilizado. O objetivo do treinamento é encontrar o conjunto de parâmetros (número de camadas, número de neurônios nas camadas e pesos entre as camadas) que minimize a diferença entre os valores de saída da rede e os valores desejados. Outro ponto a ser considerado é a estrutura da rede. Se a rede tiver uma arquitetura com muitas camadas ocultas ou for treinada por muitas épocas (a quantidade de épocas neste caso varia de acordo com os dados a rede envolvida), ela será capaz de memorizar todos os exemplos. Assim ela forma uma extensa tabela de busca, mas não irá necessariamente realizar boas generalizações para entradas que não foram vistas antes, sendo isto chamado de *overtraining.* Uma das maneiras de testar a exatidão da rede é tentar várias arquiteturas e, com a técnica de validação cruzada, verificar a melhor.

A técnica de validação cruzada, ou *k-fold-cross-validation* (*k*-FCV)*,* consiste em particionar aleatoriamente o arquivo de padrões em *k* partes de mesmo tamanho. Assim, ocorre a geração dos arquivos para treinamento e validação. As etapas de treinamento e validação são repetidas *k* vezes, sendo utilizados para treinamento *k*-1 arquivos e para validação o *k*-é*simo* arquivo não utilizado no treinamento. A cada interação, o arquivo de validação possui um *k* diferente.

Outros métodos de validação que podem ser citados são: *holdout* e *jackknife.* O método *holdout* consiste em separar, de forma aleatória, o arquivo de padrões em dois arquivos. O de treinamento tipicamente conterá dois terços dos dados e o de validação o um terço restante. Já o método *jackknife*, conhecido com *leave-one-out,* é semelhante ao *k*-FCV, mas *k* é igual ao número de linhas do arquivo de padrões. Com isto, cada arquivo de validação conterá somente uma linha em cada etapa do processo (BALDI e BRUNAK, 2001).

4.3.1 Algoritmos de aprendizado

O procedimento utilizado para realizar o processo de aprendizagem é chamado algoritmo de aprendizagem, cuja função é modificar os pesos sinápticos de forma a alcançar o objetivo desejado.

Os algoritmos de aprendizado podem ser supervisionados ou não-supervisionados, embora aspectos de cada um podem co-existir em uma dada arquitetura. O treinamento supervisionado é acompanhado pela apresentação de uma seqüência no vetor de treinamento associada com um vetor de saída alvo. Um ingrediente essencial neste tipo de aprendizado é a disponibilidade de um "professor" externo. Em termos conceituais, podemos pensar que o professor tem o conhecimento da saída desejada. O conhecimento disponível pelo professor é então transferido à RN através de ajustes interativos para minimizar o erro de acordo com o algoritmo de aprendizado (WU, 1997). Como exemplo, os algoritmos: *Back-propagation (BP), Resilient Proapagation (RProp), Cascade Correlation, Kohonen e Quickprop.* A principal diferença entre eles está, principalmente, no modo como os pesos da rede são ajustados.

Um algoritmo de aprendizado supervisionado tem o objetivo de minimizar a diferença entre o valor de saída da rede e o valor desejado. Uma típica função de erro a ser minimizada é:

$$E = \sum_{i=1}^{n} \left(y_i - h_w(x) \right)^2 \qquad \textbf{(4.6)}$$

onde *n* é o número de padrões de entrada, y_i é a saída da rede (para um dado conjunto de parâmetros- *w*) e $h_w(x)$ o valores esperado de saída. Se uma rede possui mais que uma unidade na camada de saída, então a equação 4.6 se torna:

$$E = \sum_{i=1}^{n} \sum_{j=1}^{k} \left(y_i - h_w(x) \right)^2 \qquad \textbf{(4.7)}$$

onde *k* é o número de unidades na camada de saída (WU e McLARTY, 2000).

O treinamento não-supervisionado ou aprendizado auto-organizável não possui um professor externo para verificar o processo de aprendizado. O algoritmo normalmente é guiado pela medida de similaridade sem um vetor alvo de especificação. As redes auto-organizáveis modificam os pesos até que os vetores mais similares sejam designados ao mesmo grupo de saída (clusterização), o qual é representado por um vetor-exemplo. Como exemplo de algoritmo de aprendizado não-supervisionado, pode-se ser citados os mapas auto-organizáveis de Kohonen e a teoria da ressonância adaptativa (ART) (WU, 1997).

4.3.2 Treinamento de *Perceptrons*

O princípio do treinamento de *Perceptrons* é simples: tentar um conjunto de pesos. Se a saída da RN, em resposta a um vetor de entrada, assemelha-se com o valor de saída esperado para as entradas, então segue-se ao próximo vetor de entrada. Se o valor de saída da RN for muito mais alto que o valor desejado, os pesos são diminuídos e tenta-se de novo. Se o valor for muito mais baixo, então os pesos são aumentados e o treinamento ocorre de novo. Este processo é repetido para todos os vetores de entrada até que nenhuma mudança necessite ser feita (WU e McLARTY, 2000).

Conforme Russell (2003), a idéia por trás desse algoritmo, e na verdade, por trás da maioria dos algoritmos para a aprendizagem de RNs, é ajustar os pesos da rede para minimizar alguma medida do erro no conjunto de treinamento. Desse

modo, a aprendizagem é formulada como uma busca de otimização no espaço de pesos. Conforme descrito em Russell (2003), a medida clássica é a soma dos erros quadráticos. O erro quadrático para um único exemplo de treinamento com entrada *x* e saída verdadeira *y* é escrito como:

$$E = \frac{1}{2} Err^2 \equiv \frac{1}{2}\left(y - h_w(x)\right)^2 \qquad \textbf{(4.8)}$$

onde $h_w(x)$ é a saída do perceptron.

Podemos usar o declínio do gradiente para reduzir o erro quadrático, calculando a derivada parcial de *E* em relação a cada peso. O ajuste nos pesos é dados por:

$$\Delta w(j) = \frac{-\alpha \partial E}{\partial w(j)} \qquad \textbf{(4.9)}$$

$$\frac{\partial E}{\partial W_j} = Err \times \frac{\partial Err}{\partial W_j} \qquad \textbf{(4.10)}$$

$$= Err \times \frac{\partial}{\partial W_j} g\left(y - \sum_{j=0}^{n} W_j x_j \right) \qquad \textbf{(4.11)}$$

$$= -Err \times g'(in) \times x_j \qquad \textbf{(4.12)}$$

onde α é a taxa de aprendizagem e *g'* é a derivada da função de ativação. No algoritmo de declínio do gradiente, onde se quer reduzir *E,* o peso é atualizado como a seguir:

$$Wj \leftarrow Wj + \alpha \times Err \times g'(in) \times x_j \qquad \textbf{(4.13)}$$

Intuitivamente, isso faz sentido. Se o erro $Err = y - h_w(x)$ é positivo, então a saída da rede é pequena demais e, portanto os pesos são aumentados para as

entradas positivas e diminuídos para as entradas negativas. Acontece o oposto quando o erro é negativo.

O algoritmo do aprendizado de declínio do gradiente para perceptrons, é mostrado a seguir (RUSSELL, 2003):

função APRENDIZAGEM–DE–PERCEPTRON (*exemplos, rede*) **retorna** uma hipótese de perceptrons

entrada: *exemplos*, um conjunto de exemplos, cada um com entrada **x** = $x_1,...,x_n$ e saída *y*

rede, um perceptron com pesos W_j, $j = 0...n$ e função de ativação *g*

repita

para cada *e* **em** *exemplos* **faça**

$$in \leftarrow \sum_{j=0}^{n} W_j x_j[e]$$

$$Err \leftarrow y[e] - g(in)$$

$$W_j \leftarrow W_j + \alpha \times Err \times g'(in) \times x_j x_j[e]$$

até algum critério de parada ser satisfeito

retornar HIPÓTESE–DA–REDE NEURAL (*rede)*

4.3.3 Treinamento de Redes Neurais Multicamadas

Rumelhart et al (1986) criaram um método intuitivo que aprende rapidamente, revolucionando o campo das RNs. O método foi chamado de BP porque o erro é propagado da saída para a entrada da rede, ou seja, a propagação do erro pode ser efetuada da camada de saída para a camada oculta e desta para a camada de entrada. O erro nas camadas ocultas parece misterioso, porque os dados de treinamento não informam que valor os neurônios ocultos devem ter. O processo de propagação de retorno emerge diretamente de uma derivação do gradiente de erro global e da aplicação da regra da cadeia (WU e McLARTY, 2000; RUSSELL, 2003).

Uma RN multicamadas tem três características distintas:

1. O modelo de cada neurônio da rede inclui uma *função de ativação não-linear*, como a função logística. A presença da não-linearidade é importante porque, do contrário, a relação entrada-saída da rede poderia ser reduzida àquela de perceptron de camada única.

2. A rede contém uma ou mais camadas de neurônios ocultos, que não são parte da entrada ou da saída da rede. Estes neurônios capacitam a rede a aprender tarefas complexas extraindo progressivamente as características mais significativas dos vetores de entrada.

3. A rede exibe um alto grau de conectividade, determinado pelas sinapses da rede.

Estas características que conferem o poder computacional da MLP, mas também são responsáveis pelas deficiências na compreensão do comportamento da rede (HAYKIN, 1999).

4.3.3.1 O algoritmo Back-propagation

O treinamento com o BP envolve três estágios: *(i)* o *feedforward* dos padrões dos dados de entrada do treinamento, *(ii)* o cálculo e a b*ack-propagation* do erro associado e, *(iii)* o ajuste dos pesos. Na fase *feedforward* (passo para frente), os pesos permanecem inalterados através da rede e os sinais de função da rede são computados neurônio por neurônio e um conjunto de saídas é produzido como a resposta real da rede. Na fase b*ack-propagation* (passo para trás), os sinais de erro são computados recursivamente para cada neurônio, iniciando da camada de saída e passando de trás para frente através da rede, camada por camada, para produzir o erro das unidades ocultas. Os pesos são, então, ajustados para diminuir a

diferença entre a saída obtida pela rede e a saída desejada, de acordo com uma regra de correção de erro (WU, 1997; WU e McLARTY, 2000 e HAYKIN, 1999). O algoritmo BP procura minimizar o erro obtido pela rede ajustando os pesos e limiares para que eles correspondam às coordenadas dos pontos mais baixos da superfície de erro. Para isto, ele utiliza um método de descida do gradiente.

Conforme o texto de Russell (2003), o gradiente de uma função está na direção e sentido em que a função tem taxa de variação máxima. Isto garante que a rede caminha na superfície na direção que vai reduzir mais o erro obtido. Para superfícies simples, este método certamente encontra a solução com erro mínimo. Para superfícies mais complexas, esta garantia não mais existe, podendo levar o algoritmo a convergir para mínimos locais, conforme a figura 4.7 a seguir.

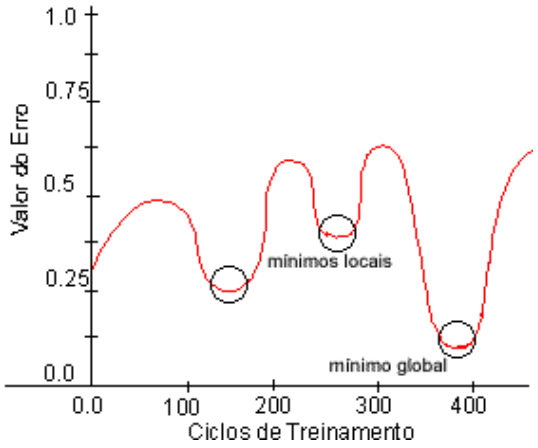

Figura 4.7 – Gráfico de uma possível superfície de erro indicando mínimos locais e mínimo global

O processo de propagação do erro no BP é muito semelhante ao descrito na seção do *Perceptron*, sendo a principal diferença que, enquanto o erro $y - h_w$ na saída é claro, o erro nas camadas ocultas parece misterioso, porque os dados de treinamento não informam que valor os neurônios ocultos devem ter. O cálculo da propagação do erro da camada de saída para as camadas ocultas emerge

diretamente do gradiente de erro global. Na camada de saída, a regra de atualização dos pesos é idêntica à Equação 4.9. Assim, no BP, os pesos são atualizados conforme a seguir:

$$W_{ji} \leftarrow W_{ji} + \Delta W_{ji} \qquad \textbf{(4.14)}$$

onde ΔW_{ji} é dado por:

$$\Delta W_{ji} = -\alpha \frac{\partial E}{\partial W_{ji}} \qquad \textbf{(4.15)}$$

A expressão $\frac{\partial E}{\partial W_{ji}}$ parte do erro quadrático, definido por:

$$E = \sum_{i=1}^{n} \left(y_i - a_i\right)^2 \qquad \textbf{(4.16)}$$

Aqui, a ativação a_i é expandida:

$$\begin{aligned} \frac{\partial E}{\partial W_{ji}} &= \left(y_i - a_i\right)\frac{\partial a_i}{\partial W_{ji}} = -\left(y_i - a_i\right)\frac{\partial g(in_i)}{\partial W_{ji}} \\ &= -\left(y_i - a_i\right)g'(in_i)\frac{\partial g(in_i)}{\partial W_{ji}} - \left(y_i - a_i\right)g'(in_i)\frac{\partial}{\partial W_{ji}}\left(\sum_j W_{ji}a_j\right) \\ &= -\left(y_i - a_i\right)g'(in_i)a_j = -a_j \times \Delta_i \end{aligned} \qquad \textbf{(4.17)}$$

Finalizando a regra de atualização dos pesos da camada de saída para a camada oculta conforme a equação a seguir:

$$W_{ji} \leftarrow W_{ji} + \alpha \times a_j \times \Delta_i \qquad \textbf{(4.18)}$$

Para atualizar as conexões entre as unidades de entrada e as unidades ocultas é efetuada a propagação de retorno do erro. A idéia é que o neurônio oculto *j* é responsável por alguma fração do erro Δ_i em cada um dos nós de saída aos quais ele se conecta. Desse modo, os valores Δ_i são divididos de acordo com a

intensidade da conexão entre o nó oculto e o nó de saída, e são propagados de volta para fornecer os valores Δ_i referentes à camada oculta.

Assim, a regra de atualização de pesos correspondente aos pesos entre as entradas e a camada oculta é quase idêntica à regra de atualização para a camada de saída:

$$W_{kj} \leftarrow W_{kj} + \Delta W_{kj} \qquad \textbf{(4.19)}$$

onde ΔW_{kj} é dado por:

$$\Delta W_{kj} = -\alpha \frac{\partial E}{\partial W_{kj}} \qquad \textbf{(4.20)}$$

e $\frac{\partial E}{\partial W_{kj}}$ origina-se da expansão das ativações a_i e a_j, conforme a seguir:

$$\begin{aligned}
\frac{\partial E}{\partial W_{kj}} &= \left(y_i - a_i\right)\frac{\partial a_i}{\partial W_{kj}} = -\sum_i \left(y_i - a_i\right)\frac{\partial g(in_i)}{\partial W_{kj}} \qquad \textbf{(4.21)} \\
&= -\sum_i \left(y_i - a_i\right) g'(in_i)\frac{\partial g(in_i)}{\partial W_{kj}} = -\sum_i \Delta_i \frac{\partial}{\partial W_{kj}}\left(\sum_j W_{ji} a_j\right) \\
&= -\sum_i \Delta_i W_{ji}\frac{\partial a_i}{\partial W_{kj}} = \sum_i \Delta_i W_{ji}\frac{\partial g(in_j)}{\partial W_{kj}} \\
&= -\sum_i \Delta_i W_{ji} g'(in_j)\frac{\partial g(in_j)}{\partial W_{kj}} \\
&= -\sum_i \Delta_i W_{ji} g'(in_j)\frac{\partial}{\partial W_{kj}}\left(\sum_k W_{kj} a_k\right) \\
&= -\sum_i \Delta_i W_{ji} g'(in_j) a_k = -a_k \times \Delta_j
\end{aligned}$$

$$\frac{\partial E}{\partial W_{kj}} = -a_k \times \Delta_j \qquad \textbf{(4.22)}$$

Onde Δ_j é definido por:

$$\Delta_j = g'(in_j)\sum_i W_{ji}\Delta_i \qquad \textbf{(4.23)}$$

Sendo, então, a regra de atualização dos referidos pesos dada pela equação:

$$W_{kj} \leftarrow W_{kj} + \alpha \times a_k \times \Delta_j \qquad \textbf{(4.24)}$$

O processo de propagação de retorno pode ser resumido desta forma:

- calcular os valores Δ para as unidades de saída, usando o erro observado.

- começando pela camada de saída, repetir as etapas a seguir para cada camada na rede, até ser alcançada a camada oculta conectada à camada de entrada: *(i)* propagar os valores Δ de volta até a camada anterior; *(ii)* atualizar os pesos entre as duas camadas.

O algoritmo BP é apresentado a seguir (RUSSELL, 2003):

Função APRENDIZAGEM–POR–PROPAGAÇÃO–DE-RETORNO (e*xemplos, rede*) **retorna** uma rede neural

Entradas: *exemplos*, um conjunto de exemplos, cada um com vetor de entrada **x** e vetor de saída *y*

rede, uma rede de várias camadas com *L* camadas, pesos W_{ji} e função de ativação *g*

Repita

Para cada *e* **em** *exemplos* **faça**

Para cada neurônio *j* na camada de entrada **faça** $a \leftarrow x_j[e]$

Para $l = 2$ **até** *M* **faça**

$in_i \leftarrow \sum_j W_{ji}\, a_j$

$a_i \leftarrow g(in_i)$

Para cada neurônio *i* na camada de saída **faça**

$\Delta_i \leftarrow g'(in_i) \times (y_i[e] - a_i)$

Para $l = M - 1$ **até** 1 **faça**

Para cada neurônio *j* na camada $l + 1$ **faça**

$$W_{ji} \leftarrow W_{ji} + \alpha \times a_j \times \Delta_i$$

Até algum critério de parada ser satisfeito
Retornar HIPÓTESE–DA –REDE NEURAL (*rede)*

O desenvolvimento do algoritmo BP representa um marco nas RNs, pois fornece um método computacional eficiente para o treinamento de MLPs. Apesar de não podermos afirmar que o algoritmo forneça uma solução ótima para todos os problemas resolúveis, ele acabou com o pessimismo sobre a aprendizagem em máquinas de múltiplas camadas.

4.4 EXTRAÇÃO DE REGRAS A PARTIR DE REDES NEURAIS

Uma das características mais atrativas das RNs é que elas não requerem um conhecimento prévio da aplicação do problema para a construção do modelo. Assim, para tornar a tecnologia de RNs realmente compreensível ao usuário, é desejável extrair conhecimento a partir de redes treinadas e descobrir regras biológicas essenciais (WU e MCLARTY, 2000). Muitas vezes, as RNs são denominadas de "caixa preta", em particular por fornecer ao usuário nenhuma informação sobre o conhecimento adquirido. Embora isto geralmente seja verdadeiro, especialmente para redes multicamadas, para as quais os pesos não podem ser facilmente interpretados, existem métodos para analisar RNs e extrair regras ou características. Estas regras incluem: regras de inferência (*if-then-else)*, árvores de decisão, regras difusas, entre outras.

Apesar da classe de regras geradas, algumas características devem ser buscadas, conforme descrito em Tickle et al (1998): *(i)* poder expressivo ou formato da regra; *(ii)* qualidade; *(iii)* translucidez; *(iv)* complexidade algorítmica da regra ou da técnica de extração das regras e *(v)* portabilidade.

Por **poder expressivo** sugere-se três grupos de formatos de regras: regras simbólicas convencionais (Booleana, proposicional), regras baseadas em lógica *fuzzy* e as regras expressas na forma de lógica de primeira ordem. A **qualidade** da regra é dada por um conjunto de quatro medidas: *(i)* acurácia (o grau com o qual o conjunto de regras extraídas é capaz de classificar exemplos "não-vistos" de forma correta); *(ii)* fidelidade (grau de similaridade entre as regras extraídas e a RN da qual se originaram); *(iii)* consistência (indica o grau com que, sob diferentes treinamentos, a RN gera regras que produzam a mesma classificação para os casos "não vistos") e *(iv)* compreensibilidade (tamanho do conjunto de regras extraídas - número de regras e de antecedente por regra) (ARBATLI e AKIN, 1997; ANDREWS et al, 1995; TICKLE et al, 1998).

O critério de **translucidez** busca categorizar uma técnica de extração de regras baseado na granularidade da RN, a qual pode ser implícita ou explicita. Conforme Andrews et al (1995), existe três identificadores chave (decomposicional, eclética e pedagógica), para definir pontos de referência no espectro de tais níveis de granularidade percebidos. Estes indicadores de regras serão especificados na subseção 4.4.1, a seguir.

A **complexidade do algoritmo** imobiliza a técnica de extração de regras. Entretanto, como descrito em Andrews et al (1995), comparações nesta área são impedidas pelo fato de que um número substancial de autores não reportam ou comentam este assunto. A portabilidade define a extensão de que uma dada técnica possa ser aplicada através de um grupo de arquiteturas de RNs e regimes treinados. A necessidade para este critério foi estabelecido com base na característica das técnicas para extração de regras a partir de RNs treinadas foi a preponderância

Conforme Andrews et al (1995), a extração de regras pode oferecer alguns benefícios listados a seguir:

- descoberta de novos relacionamentos e/ou características importantes a partir das regras extraídas;

- expressão do conhecimento de modo formal;

- capacidade de gerar explicações para as decisões tomadas internamente pela RN, de modo que facilite a aceitação do uso da rede pelos usuários;

- integração com sistemas simbólicos e, assim, a possibilidade de descobrir em que situações a rede pode cometer erros de generalização;

- identificação de regiões no espaço de entrada que não se fizeram representar no conjunto de treinamento.

Além disso, as regras extraídas a partir das RNs, podem ser apresentadas para um especialista, que as analisa e verifica se há incorreções. Assim, as regras corretas podem ser usadas para gerar padrões de treinamento adequados, os quais podem melhorar a capacidade de generalização da rede (CLOETE e ZURADA, 2000). Uma vez que este trabalho visa extrair regras a partir de RNs, a próxima seção descreve brevemente alguns tipos de regras, com ênfase maior às regras do tipo *if-then*.

4.4.1 Tipos de regras

A extração de regras a partir de RNs é baseada no comportamento dos neurônios, sendo a relação entre as entradas e as saídas usualmente analisada (CLOETE e ZURADA, 2000; HUANG e XING, 2002).

Conforme Andrews et al (1995), há muitos tipos de regras que podem ser extraídos das RNs, mas o desenvolvimento de técnicas de extração de regras tem sido mais direcionado à apresentação da saída como um conjunto de regras expressas, usando a forma convencional de lógica simbólica na forma *if...then...else....*Muitos trabalhos têm direcionado seus esforços para a extração de regras utilizando a lógica *fuzzy*. Esta também permite que as regras sejam expressas na estrutura *if...then...else...* . Contudo, elas utilizam-se do conceito de funções de pertinência para tratar quais são os termos parcialmente verdadeiros, por exemplo, *se x é baixo e y é alto, então z é médio*, onde *baixo*, *alto* e *médio* são conjuntos *fuzzy* com correspondência às funções.

Neste tipo de regra, a parte SE especifica um conjunto de condições sobre valores de atributos previsores e a parte ENTÃO especifica um valor previsto para o atributo de saída. Os atributos previsores são as premissas da regra que devem ser obedecidas, para assim obter um atributo classe.

IF < condição> *THEN* <conclusão> (<confidência>)

A “condição” é, tipicamente, uma expressão lógica que contém variáveis relevantes das quais os valores podem ser inferidos a partir das bases de fatos ou fornecidos pelo usuário. A “conclusão” determina o valor de alguma variável que corresponde para a “condição” ser satisfeita. O grau de certeza ou validade da regra é expressa pelo seu percentual de confidência (CLOETE e ZURADA, 2000). A extração de regras é realizada através da interpretação dos pesos da rede neural.

As regras do tipo *if-then* podem ser utilizadas posteriormente em um sistema de inferência lógica para a resolução de problemas. Um segundo uso destas regras pode ser a geração de regras para um sistema baseado em conhecimento. Deve-se

observar, também, que quanto mais curtas as regras (em termos de números de cláusulas) melhor, pois regras curtas geralmente podem ser aplicadas a mais situações (CLOETE e ZURADA, 2000).

Segundo Andrews et al (1995), a extração de regras podem ser feitas por duas categorias básicas de técnicas, uma chamada de **técnica decomposicional** e outra de **técnica pedagógica**. Existe ainda uma terceira técnica, chamada de **eclética**, a qual combina elementos das duas categorias básicas. Um refinamento adicional nesta nomenclatura foi descrita em Tickle et al (1998), que propôs o termo de **técnica composicional** para a extração de regras a partir de um conjunto de neurônios, mais do que dos neurônios individualmente.

A principal característica da técnica decomposicional é que o foco da extração de regras está no nível de unidades individuais (camada oculta e saída). Assim, a estrutura da rede é a principal fonte de regras. Um requerimento básico para a extração de regras desta abordagem é que a saída computada por cada unidade oculta e de saída pode ser mapeada em um resultado binário (sim/não). Já a idéia central da técnica pedagógica, está na visão de que a extração de regras é como uma tarefa de aprendizado, na qual o conceito alvo é a função computada pela rede e as características de entrada são simplesmente as características das entradas da rede. Portanto, esta técnica objetiva a extração de regras que mapeiam as entradas diretamente com as saídas, sem se preocupar com os passos intermediários (ANDREWS et al, 1995).

A técnica eclética inclui abordagens que utilizam o conhecimento sobre a arquitetura interna e/ou vetores de pesos para complementar um algoritmo de aprendizado simbólico que utiliza dados de treinamento (ANDREWS et al, 1995). A

técnica composicional, tem como exemplo a extração de autômatos de estado finito determinístico de redes neurais recorrentes. Esta técnica analisa o estado do espaço consistindo de todos os neurônios recorrentes de modo a encontrar agrupamentos de ativações de neurônios ocultos. Portanto, a técnica não é estritamente decomposicional porque não extrai regras de neurônios individuais com subseqüente agregação para formar uma relação global; não pode ser eclética porque não há aspecto que o enquadre no perfil pedagógico (TICKLE et al, 1998).

4.4.2 Regras obtidas a partir dos neurônios da camada oculta

Para a obtenção de regras a partir dos neurônios da camada oculta da RN treinada, o programa denominado FAGNIS (CECHIN,1998), analisa o valor de ativação dos neurônios na camada oculta e os classifica em três regiões, conforme ilustrado na Figura 4.8. Para cada entrada da rede, verifica-se em qual das regiões a ativação dos neurônios ocultos se enquadram. O número máximo de combinações possíveis é 3^n, onde *n* simboliza o número de neurônios na camada oculta. No entanto, nem todas estas combinações ocorrem e, somente as combinações mais freqüentes são consideradas, porque elas representam melhor os dados. Como resultado, temos o protótipo da regra. Por protótipo, definimos a média das entradas de cada grupo (combinação das regiões). Assim, a escrita formal da regra possui a forma de uma equação linear: "SE $X \cong$ protótipo ENTÃO Y = constante da equação linear + (os coeficientes da equação linear) * X." Aqui, X é o exemplo de entrada; Y corresponde à saída da RN e os coeficientes da equação linear representam a influência dos exemplos na saída da RN.

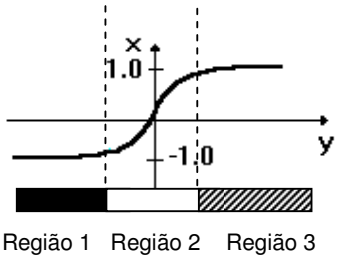

Figura 4.8 – Ilustração das três regiões definidas na função sigmóide para análise dos dados de entrada e extração de regras.

4.4.3 Árvores de Decisão

As árvores são ferramentas analíticas usadas para descobrir regras e relações pela subdivisão da informação contida no conjunto de dados analisados (WESTPHAL e BLAXTON, 1998). Elas consistem de **nodos** que representam os atributos, de **arcos** provenientes destes nodos e que recebem os valores possíveis para estes atributos. O primeiro nodo é chamado de raiz, pois dele derivam os outros nodos, chamados de folhas, que representam as diferentes classes de um conjunto de treinamento. As árvores de decisão podem ser representadas, também, por conjuntos de regras *if-then*, já que estas são mais legíveis. Cada regra representa um possível caminho a ser percorrido desde a raiz até uma folha, onde o resultado da classificação é especificado.

Uma árvore de decisão tem a função de particionar recursivamente um conjunto de treinamento, até que cada subconjunto obtido deste particionamento contenha casos de uma única classe. Para atingir esta meta, esta técnica examina e compara a distribuição de classes durante a construção da árvore. O resultado obtido (após a construção da árvore) são os dados organizados de maneira compacta, que são utilizados para classificar novos casos. Portanto, as árvores de

decisão podem ser consideradas um meio eficiente de construir classificadores que predizem classes baseadas nos valores de atributos de um conjunto de dados (MITCHELL,1997). Um exemplo de árvore de decisão pode ser visualizado na Figura 4.9, apresentada a seguir.

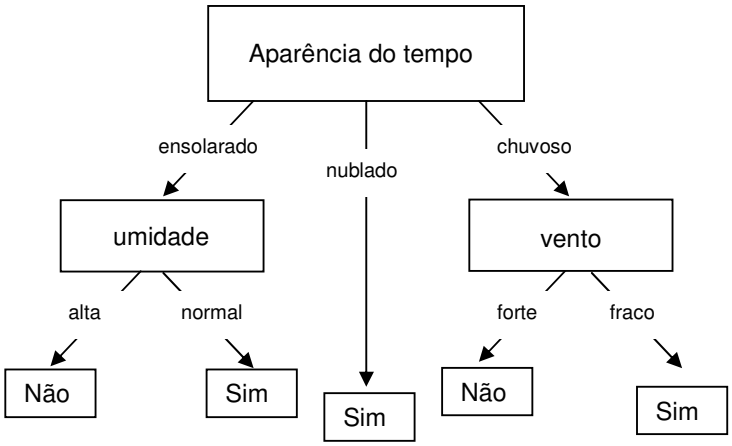

Figura 4.9 – Árvore de decisão para a escolha se o dia é apropriado ou não para jogar tênis. (MITCHELL, 1997).

O exemplo acima ilustra uma árvore de decisão para decidir sobre dias adequados para se jogar tênis, de acordo com os atributos: aparência do tempo, temperatura, umidade e força do vento. A decisão por jogar tênis pode ser expressa sob a forma de disjunções de conjunções (como mostrado abaixo), sendo que cada conjunção representa um caminho na árvore desde o nodo raiz até a folha.

Jogartênis = (aparência = ensolarado e umidade = normal) ou (aparência = nublado) ou (aparência = chuvoso e vento = fraco)

Após os conceitos fundamentais de RNs e de extração de regras, os próximos capítulos descrevem a metodologia empregada e os resultados obtidos, bem como a discussão destes.

5 METODOLOGIA

Nesta seção, os elementos da metodologia são apresentados individualmente. Foram realizadas duas simulações diferentes para o problema proposto: uma simulação utilizou a seqüência promotora codificada em uma base ortogonal e a outra utilizou a informação de estabilidade da seqüência promotora. Para simplificação, será denominada de **Simulação 1** aquela que utiliza a informação com codificação ortogonal e **Simulação 2**, aquela que utiliza os valores de estabilidade das seqüências.

5.1 ORGANISMO ESTUDADO

O organismo escolhido foi a bactéria *Escherichia coli* porque esta possui a maior quantidade de dados experimentais de alta qualidade disponíveis em bancos de dados públicos.

5.2 BANCO DE DADOS

As regiões promotoras de *E. coli* e os dados relacionados à estrutura e expressão de genes foram retirados de bancos de dados biológicos públicos. Os bancos utilizados foram:

- **RegulonDB:** base de dados de domínio público, que contém informações sobre a rede regulatória de *E. coli* com conhecimento experimental. Há dados sobre a organização de operons, promotores e fatores sigma associados, entre outros (SALGADO et al, 2006). As informações estão disponíveis no endereço de internet: http://regulondb.ccg.unam.mx/index.html.

Os bancos de dados descritos a seguir foram utilizados eventualmente para a análise mais completa dos dados, esclarecimento de dúvidas e para obtenção de outras informações importantes para a compreensão e análise dos resultados. Em cada um deles, há uma descrição de que tipo de dado foi obtido.

- **NCBI:** maior base pública de dados de seqüências genéticas. Nela há seqüências de genes, proteínas, genomas completos, dados de homologia e expressão gênica, além de possuir informações sobre os artigos relacionados a cada descoberta genética (WHEELER et al, 2006). As informações estão disponíveis no endereço de internet: http://www.ncbi.nlm.nih.gov.

- **KEGG:** base de dados pública de vias metabólicas e informações sobre genes e expressão gênica (OGATA et al, 1999). As informações estão disponíveis no endereço de internet: http://www.genome.jp/kegg.

- **EcoCyc:** banco de dados público com diferentes categorias de informações sobre *E. coli*. Aqui há mapas metabólicos, organização gênica, regulação transcricional, entre outras (KESELER et al, 2005). As informações estão disponíveis no endereço de internet: http://ecocyc.org.

5.3 FERRAMENTAS

As principais ferramentas computacionais utilizadas foram:

- **ClustalW:** software de domínio público que realiza alinhamento de seqüências biológicas (THOMPSON et al*,* 1994; 1997). Ele pode ser acessado localmente ou pelo site http://align.genome.jp.

- **Python:** linguagem de programação escolhida para desenvolver programas para a automatização da preparação de dados para as etapas de treinamento e teste (codificação ortogonal e da estabilidade) e organização dos arquivos na metodologia *ten-fold cross validation.* Também auxiliou na análise estatística dos resultados (PYTHON SOFTWARE FOUNDATION, 2006).

- **R:** software de domínio público para manipulação e análise de dados. Este software permite a realização de análise estatística, treinamento de RNs, extração de regras, entre outras funções (R DEVELOPMENT CORE TEAM).

- **SPSS**: software para análise estatística. Permite a criação de gráficos e outras funções (SPSS).

- **Tisean:** software de domínio público que realiza a suavização de dados através de um filtro passa-baixa – LowPass – (HEGGER et al, 1999).

- **WEKA:** software de domínio público (Java), desenvolvido pela Universidade de Waikato, contém uma série de algoritmos de *data mining* (WITTEN e FRANK, 2005).

5.4 PREPARAÇÃO DOS DADOS

O primeiro passo em uma metodologia de AM é reunir os exemplos para a etapa de treinamento. Foi escolhida a base de dados RegulonDB foi a que apresentou o maior número de seqüências e, também, o maior número de

informações sobre estas. Assim, as regiões promotoras foram retiradas deste banco de dados em sua versão disponível em janeiro de 2006. As seqüências possuíam tamanho de 81 nucleotídeos, sendo que aproximadamente os vinte nucleotídeos finais da seqüência estavam na região *downstream*, ou seja, após o TSS.

Utilizou-se 940 seqüências promotoras e 940 seqüências aleatórias para representar os exemplos negativos. Estas foram embaralhadas e divididas em 10 arquivos para constituir os conjuntos de treinamento e teste. A primeira escolha para os exemplos negativos foram seqüências gênicas. No entanto, estas possuem certas regularidades que eram aprendidas pela rede, ocasionando um falso aprendizado; já que a rede aprendia padrões das seqüências não-promotoras e os utilizava como parâmetros de classificação. Assim, as seqüências aleatórias mostraram-se mais apropriadas. Os exemplos negativos foram gerados com o auxílio de um gerador de seqüências aleatórias com probabilidade de 0,25 para cada nucleotídeo. Esta probabilidade é a mesma encontrada nas seqüências promotoras.

5.4.1 Preparação dos Dados para realização da Simulação 1

Para a realização do treinamento na **Simulação 1** as seqüências foram representadas por uma codificação ortogonal de quatro dígitos binários dados por: A= 0100, T= 1000, C= 0001 e G= 0010, conforme Demeler e Zhou (1991). Inicialmente, a rede foi treinada com as seqüências sem um alinhamento prévio. Após, realizou-se simulações com os promotores previamente alinhados, por meio da utilização do software ClustalW, para garantir o sincronismo nos dados de entrada. Sem o alinhamento, a RN não consegue realizar uma boa classificação.

A falta de sincronismo deve-se à posição dos hexâmeros consensuais - estes não estão em lugares fixos em todas as seqüências, já que a quantidade de nucleotídeos entre os hexâmeros -10 e -35 e entre o hexâmero -10 e o TSS é variável de um promotor para outro. A Figura 5.1 mostra um exemplo de alinhamento de seqüências de DNA.

----------CATCTATCATCTAAAAAACC--AGAAAAACAAATAAC-ATCATGT	hycA
------TTAAAAATCTCTTTAATAACAATAAAT--TAAAAGTTGGCACAA-AAAATGC	rpsUp3
---------CGGTGCTTTACAAAGCAGCAGCAATTGCAGTAAAATTCCGCACCATTTTGA	fusAp
----------GATAAATCCATGGCTCTGCGCCTGGCGAACGAACTTTCTGATGCTGCA	gugpp2
----------TTCCCTCACCCACGCCGTCACCGCCTTGTCATCTTTCTGACACCT	purH

Figura 5.1 – Representação do alinhamento de seqüências promotoras. O símbolo '-' significa *gap*.

A seqüência resultante do alinhamento utilizada como entrada na RN teve 72 nucleotídeos. Este número deve-se ao fato de que o processo de alinhamento insere *gaps* (representado pelo símbolo '-') não só no meio das seqüências alinhadas, mas também no início e no fim, formando um padrão. Para evitar falso aprendizado, tomou-se o cuidado de remover estes *gaps* do início e do fim do exemplo e os inseridos no meio da seqüência foram substituídos por '0000', conforme também realizado por Kalate et al (2003).

5.4.2 Preparação dos Dados para realização da Simulação 2

A estabilidade da molécula de DNA pode ser expressa como energia livre. Esta estabilidade depende tanto da composição dos nucleotídeos individualmente quanto da composição dos dinucleotídeos da molécula. É possível predizer a estabilidade de uma molécula de DNA a partir de sua seqüência se a contribuição de cada vizinho é conhecida. Esta informação pode ser obtida em trabalhos como os de

SantaLucia (1998) e SantaLucia et al (2004) que propõem uma maneira unificada para o cálculo da estabilidade e explicam a metodologia do próximo vizinho.

Apesar de a estabilidade ser conhecida como uma característica física da seqüência promotora, esta informação não é utilizada como parâmetro de aprendizado em estudos de promotores utilizando técnicas de AM. Até o término da revisão bibliográfica, nenhum artigo foi encontrado empregando estes dados.

Para a realização das simulações empregando a informação da estabilidade, o cálculo da energia livre se dá pela aplicação da seguinte fórmula:

$$\Delta G^{0} = \Delta G^{0}_{ini} + \Delta G^{0}_{sym} + \sum_{i=1}^{n-1} \Delta G^{0}_{ij+1} \qquad (\mathbf{5.1})$$

onde,

ΔG^{0}_{ini} é a iniciação da energia livre para os dinucleotídeos e recebe valor +1.96 kcal/mol.

ΔG^{0}_{sym} recebe valor +0.43 kcal/mol e é aplicado se a fita-dupla é auto-complementar.

ΔG^{0}_{ij} é a variação de energia livre para os dinucleotídeos de tipo *ij*.

n é o nucleotídeo da seqüência promotora.

Em nossa análise, os termos ΔG^{0}_{sym} e ΔG^{0}_{ini} não foram considerados, conforme realizado também por Kanhere e Bansal (2005b). Para calcular o valor de *ΔG*, a equação 5.1 foi aplicada em uma janela deslizante que percorreu toda a seqüência promotora. Esta janela se moveu com um incremento de um nucleotídeo por vez.

Os dados de estabilidade para os promotores mostram-se bastante irregulares, com subidas e caídas bruscas, conforme a Figura 5.2. Assim, para diminuir estas irregularidades e, visando aumentar o sincronismo dos dados, pensou-se inicialmente na aplicação de uma *wavelet.* No entanto, ela foi descartada devido à reduzida quantidade de valores da seqüência (80 valores), optando-se pela suavização dos dados com um filtro passa-baixa (LowPass).

A suavização é o processo de calcular a distribuição sobre os estados anteriores, dada a evidência até o presente. O software TISEAN (HEGGER et al, 1999), de domínio público, foi escolhido para este fim.

(a)

-0.88 -1.44 -1.3 -1.0 -1.3 -1.84 -1.3 -1.45 -1.28 -0.58 -1.3 -2.24 -1.28 -0.58 -1.0 -1.0 -1.0 -1.45 -1.84 -1.84 -1.45 -1.0 -1.0 -1.45 -1.28 -0.58 -1.3 -1.44 -1.0 -1.0 -1.44 -1.44 -1.0 -1.44 -2.24 -1.45 -1.0 -0.88 -1.0 -0.58 -1.3 -2.24 -1.45 -0.88 -1.28 -1.84 -1.28 -1.0 -1.44 -2.24 -1.45 -1.45 -1.84 -1.28 -1.28 -1.28 -0.58 -0.88 -1.44 -1.44 -0.58 -1.0 -1.0 -1.3 -1.84 -1.84 -2.24 -1.28 -1.44 -1.3 -1.0 -1.3 -1.84 -1.84 -1.3 -0.88 -1.0 -1.28 -2.17 -1.84

(b)

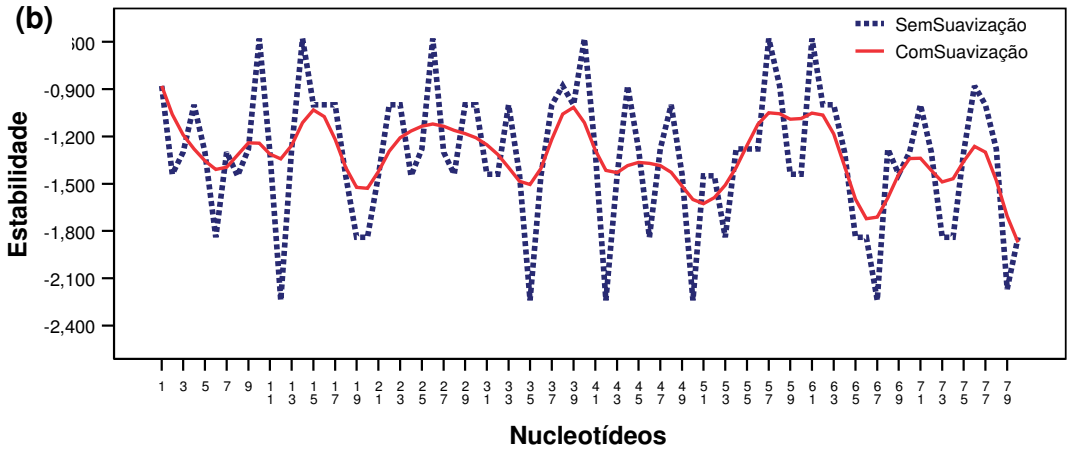

Figura 5.2 – Gráfico da estabilidade de seqüências promotoras. Em (a) Os valores de estabilidade não suavizados utilizados na como entrada para a RN. Em (b) o gráfico que compara os valores de estabilidade das seqüências. Na linha pontilhada, valores de estabilidade das seqüências promotoras sem suavização e na linha contínua a mesma seqüência com os valores suavizados em grau 5 .

Um filtro passa-baixa é um dispositivo que permite a passagem das freqüências de certa faixa e atenua as freqüências fora dessa faixa. O filtro emprega a equação a seguir:

$$x'_n=(x_{n-1}+2x_n+x_{n+1})/4 \qquad (5.2)$$

Assim, houve simulações com os dados não-suavizados e com os dados suavizados, a fim de comparar os resultados e inferir melhor sobre a caracterização das seqüências promotoras.

As próximas seções apresentam a metodologia empregada para a determinação da melhor arquitetura para as duas simulações (com codificação ortogonal e com informações de estabilidade) e quais os parâmetros utilizados para analisar a classificação das redes treinadas.

5.5 TREINAMENTO DAS REDES NEURAIS

As simulações para determinação da arquitetura que melhor classifica as seqüências foram realizadas no ambiente R, com mudanças no número de neurônios na camada de entrada e na camada oculta. A mudança no número de neurônios na camada de entrada se fará após a análise dos pesos da rede e da exatidão obtida. O algoritmo escolhido para a classificação das seqüências foi o *back-propagation*. Para a obtenção de dados estatisticamente válidos, optou-se pela metodologia de 10-*fold-cross-validation.* Assim, para cada arquitetura de RN foram realizadas 100 simulações: para cada *fold* foram realizadas 10 simulações com diferentes inicializações dos pesos da RN.

5.5.1 Determinação da melhor arquitetura para a Simulação 1

A escolha da arquitetura que melhor caracteriza as regiões promotoras utilizando a seqüência de nucleotídeos foi determinada de modo heurístico, ou seja, através da metodologia de tentativa e erro. Inicialmente, realizaram-se simulações com as seqüências promotoras não-alinhadas. O número de neurônios na camada de entrada é obtido pela quantidade de nucleotídeos multiplicada por quatro (quantidade de dígitos da codificação ortogonal), a quantidade de neurônios na camada oculta variou de acordo com o indicado na Tabela 5.1, e como saída da RN, o número de neurônios nesta camada foi sempre fixo: 1.

Após esta simulação, para melhor análise dos resultados, foram treinadas arquiteturas com os nucleotídeos *downstream* removidos, por entendermos que esta informação não auxiliava no aprendizado da rede. Portanto, houve treinamentos com seqüências de 71 nucleotídeos (10 nucleotídeos *downstream* removidos) e 61 nucleotídeos (20 nucleotídeos *downstream* removidos); além da simulação com a seqüência inteira (81 nucleotídeos). O treinamento ocorreu com até 500 épocas de aprendizado e envolveu as arquiteturas apresentadas na Tabela 5.1, a seguir.

Tabela 5.1 – Diferentes arquiteturas treinadas para predição e reconhecimento de promotores. Nestas simulações, as seqüências não estavam alinhadas.

Neurônios de Entrada	Número de Neurônios na Camada	Neurônios na Camada Saída
324	1, 5, 10, 15, 20	1
284	1, 5, 10, 15	1
244	1, 3, 5, 10	1

Após estas simulações e suas respectivas análises, as RNs foram treinadas com as seqüências previamente alinhadas pelo programa ClustalW. Para as simulações com os dados alinhados, o número de neurônios de entradas, portanto, permaneceu fixo: 288 (72 pb multiplicados pela quantidade de dígitos da codificação

ortogonal). O número de neurônios na camada de saída também foi fixo: 1. A variação ocorreu com o número de neurônios na camada oculta: houve simulações com 1 até 5 neurônios.

5.5.2 Determinação da melhor arquitetura para a Simulação 2

Para a realização da **Simulação 2**, os valores de estabilidade da seqüência foram os parâmetros de entrada da RN. A fórmula foi aplicada em diferentes tamanhos de janela, a fim de encontrar um tamanho ótimo para a classificação. A variação no número de neurônios na camada de entrada variou de acordo com o tamanho da janela, já que a seqüência promotora provinda do banco de dados tinha tamanho de 81 pb. Assim, uma janela de tamanho 2 resultou em uma seqüência de 80 valores; já uma janela de tamanho 15, resultou em uma seqüência com 67 valores de ΔG. As diferentes arquiteturas treinadas estão apresentadas na Tabela 5.2, a seguir.

Tabela 5.2 – Diferentes arquiteturas treinadas para predição e reconhecimento de promotores. Aqui, foram empregados os valores numéricos de estabilidade.

Neurônios de Entrada	Número de pb da janela aplicada na fórmula 5.1	Número de Neurônios na Camada Oculta	Neurônios na Camada Saída
80	2	1, 2, 3, 4, 5, 6, 7, 8	1
79	3	1, 2, 3, 4, 5, 6, 7, 8	1
78	4	1, 2, 3, 4, 5, 6, 7, 8	1
76	6	1, 2, 3, 4, 5, 6, 7, 8	1
72	10	1, 2, 3, 4, 5, 6, 7, 8	1
68	14	1, 2, 3, 4, 5, 6, 7, 8	1
67	15	1, 2, 3, 4, 5, 6, 7, 8	1

Após a realização destas simulações, houve simulações com os dados suavizados com o filtro passa-baixa. Este programa permite que os dados sejam suavizados em diferentes níveis, que variam de 1 até 20. A Tabela 5.3 mostra os

graus de suavização utilizados neste trabalho de dissertação. A arquitetura treinada com os dados suavizados foi: 80 neurônios na camada de entrada e 1 neurônio na camada de saída. A variação ocorreu no número de neurônios na camada oculta: de 1 até 8 neurônios (ver Tabela 5.3). Optou-se pelos valores de estabilidade provenientes de uma janela com 2 pb porque verificamos que esta, por ter maior quantidade de valores, representa melhor a diferença que existe entre as regiões *upstream* e *downstream*.

Tabela 5.3 – Diferentes arquiteturas treinadas para predição e reconhecimento de promotores. Aqui, foram empregados os valores numéricos de estabilidade suavizados.

Neurônios de Entrada	Grau de Suavização	Neurônios na Camada Oculta	Neurônios na Camada Saída
80	3	1, 2, 3, 4, 5, 6, 7, 8	1
80	5	1, 2, 3, 4, 5, 6, 7, 8	1
80	7	1, 2, 3, 4, 5, 6, 7, 8	1
80	9	1, 2, 3, 4, 5, 6, 7, 8	1
80	15	1, 2, 3, 4, 5, 6, 7, 8	1
80	20	1, 2, 3, 4, 5, 6, 7, 8	1

5.6 ANÁLISE DA CLASSIFICAÇÃO

A exatidão média da classificação obtida pela RN pode ser calculada a partir do número de verdadeiros positivos (VP), verdadeiros negativos (VN), falsos positivos (FP) e falsos negativos (FN). Entende-se por VP as seqüências promotoras reconhecidas como promotoras; por VN, as seqüências aleatórias classificadas como aleatórias; por FP, as seqüências aleatórias classificadas como promotores e por FN, os promotores identificados como seqüências aleatórias (HOWARD e BENSON,2003).

O cálculo da exatidão é dado por:

$$A = \frac{VP + VN}{VN + VP + FN + FP}. \quad \textbf{(5.3)}$$

Além da exatidão, foram analisadas a sensibilidade e a especificidade das RNs treinadas. A sensibilidade trata do índice de VP e é dada pela fórmula 5.4 e a especificidade informa sobre o índice de VN e é calculada conforme a fórmula 5.5. As fórmulas são apresentadas a seguir:

$$SN = \frac{VP}{VP + FN}. \quad \textbf{(5.4)}$$

$$S = \frac{VN}{VN + FP}. \quad \textbf{(5.5)}$$

Estes valores são importantes para a determinação da curva ROC ("*receiver operating characteristic*"). Trata-se de um gráfico que mede a taxa de falsos positivos no eixo *x* e a taxa de falsos negativos no eixo *y*, para vários pontos de compromisso (RUSSELL, 2003). Uma curva ROC é necessariamente crescente porque reflete a relação entre os valores de especificidade e sensibilidade. Cada ponto na figura representa um par especificidade/sensibilidade. Assim, constitui uma descrição empírica simples, todavia completa, que permite estudar a variação da sensibilidade e especificidade, para diferentes valores de corte, ou seja, para todas as combinações possíveis de decisões corretas e incorretas. Os dados desta curva são obtidos a partir dos resultados da classificação da RN com diferentes valores de corte, isto é, os resultados da matriz de confusão são analisados com valores de corte que variam de 0,0 até 1,0.

Um gráfico com discriminação perfeita tem uma curva que passa na esquina superior esquerda, já que nesta região é onde a fração dos VP é aproximadamente 100% e dos FP é aproximadamente 0%. A figura 5.3 mostra uma curva ROC ideal.

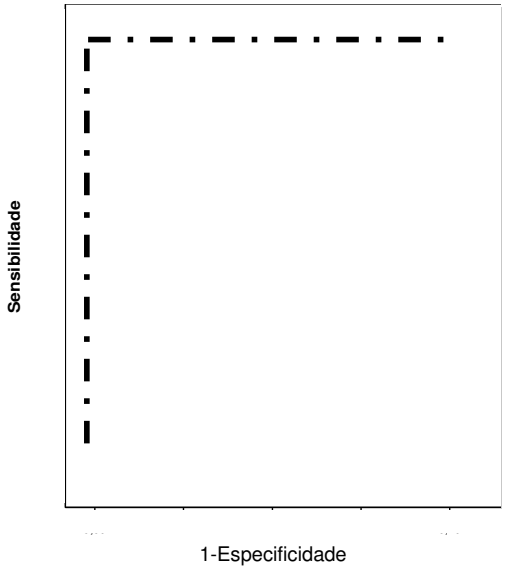

Figura 5.3 – Modelo de curva ROC ideal.

A próxima seção descreve as abordagens de extração de regras utilizadas para a concretização dos objetivos.

5.7 EXTRAÇÃO DE REGRAS

Conforme visto no capítulo 4, a metodologia de RNs possui uma grande aplicabilidade nos mais diversos problemas, mas uma de suas desvantagens é que o conhecimento adquirido por elas não é diretamente acessível. Considerando isto, tornou-se necessário o uso de uma metodologia de extração de regras das RNs treinadas as quais podem fornecer informações que auxiliem na verificação de quais regiões e/ou nucleotídeos são determinantes na aprendizagem pela rede e, permitem, asssim, comparar seu aprendizado com o conhecimento biológico já pré-estabelecido para as seqüências promotoras.

As regras extraídas serão do tipo *if ... then*. Para isto, serão empregadas duas ferramentas: *(i)* no ambiente R, por meio de um programa realizado nesta linguagem denominado FAGNIS e *(ii)* pelo software Weka, com a utilização do algoritmo J-48. No primeiro caso, a extração de regras será baseada nos valores de ativação dos neurônios da camada escondida, agrupando os resultados conforme a pontuação obtida pelos exemplos de treinamento (conforme visto na subseção 4.4.2). Já com o algoritmo J-48, que emprega árvores de decisão (conforme visto na subseção 4.4.3), pretende-se estabelecer relações entre a seqüência de entrada e a saída obtida pela rede neural, desconsiderando os pesos.

Ao empregar duas metodologias, procuramos ampliar o nosso entendimento sobre os mecanismos utilizados pela RN para o reconhecimento dos promotores e, através das regras, sistematizar e validar o aprendizado.

5.8 ESTRUTURA DA METODOLOGIA

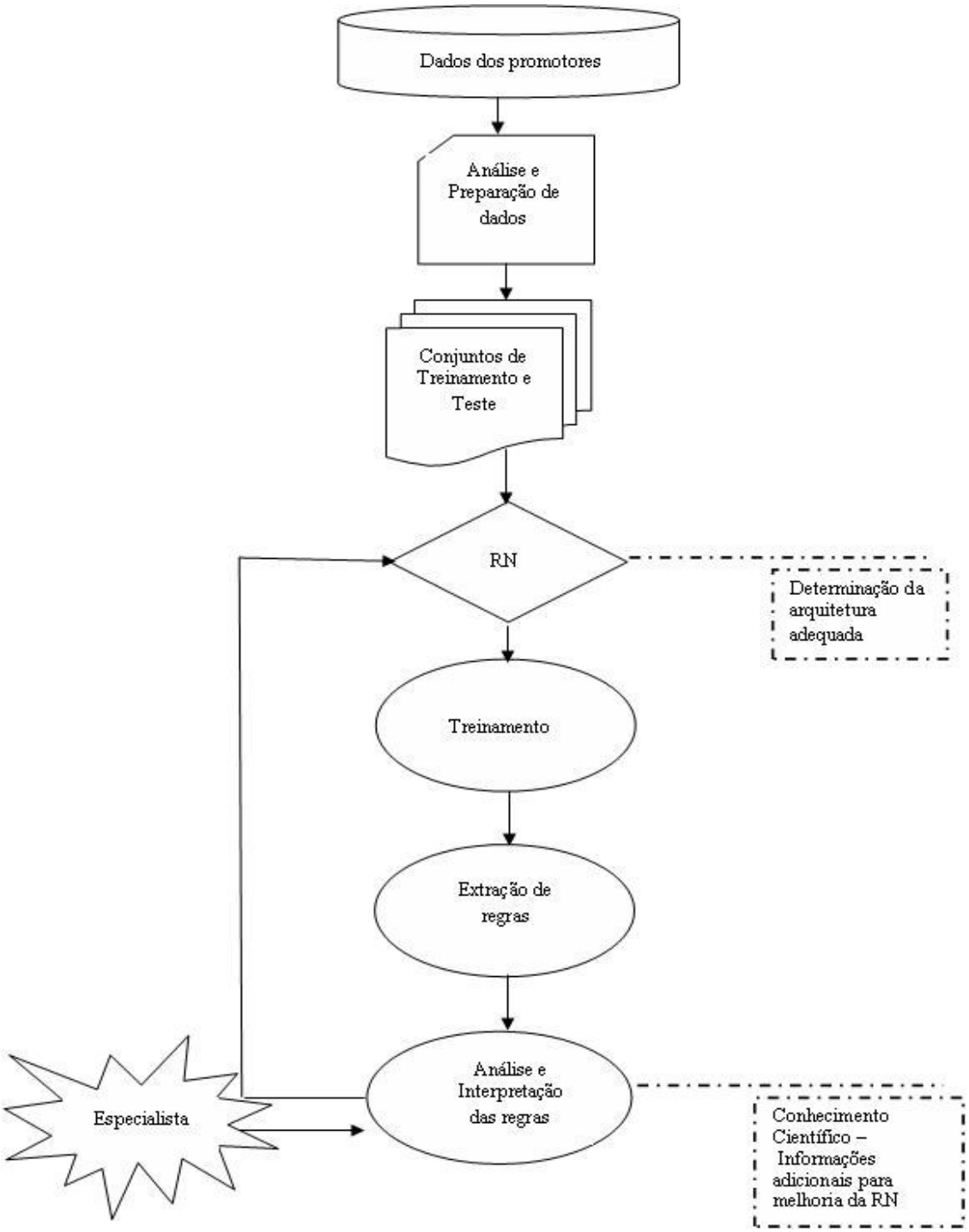

Figura 5.4 – Estrutura da metodologia proposta para o uso de RN no reconhecimento e predição de promotores.

6 RESULTADOS E DISCUSSÃO

A revisão bibliográfica nos permitiu a escolha das RNs como a metodologia que melhor possibilita a incorporação de informações físicas da molécula de DNA. Este capítulo apresenta as melhores arquiteturas para as simulações descritas no capitulo anterior, a análise da classificação e suas respectivas regras. Para a **Simulação 1** treinou-se 18 arquiteturas diferentes, o que totalizou 1800 simulações. Já para a **Simulação 2**, treinou-se 104 arquiteturas diferentes, resultando em 10400 simulações. Assim, para a análise e validação dos resultados, este número de simulações é estatisticamente significativo.

Optou-se por mostrar, ao longo das seções e subseções, apenas os resultados para a arquitetura ótima das **Simulações 1 e 2**. Isto objetiva simplificar a leitura e compreensão dos resultados. Os resultados não mostrados, mas comentados neste capítulo, encontram-se nos anexos.

6.1 MODELO DE REDE NEURAL PARA RECONHECIMENTO DE SEQÜÊNCIAS PROMOTORAS

6.1.1 Determinação da melhor arquitetura para a Simulação 1

Nesta seção, são mostrados os resultados obtidos com as redes treinadas com as seqüências não-alinhadas e alinhadas. A Tabela 6.1 apresenta o erro médio

quadrático (RMS) obtido na melhor época de cada arquitetura treinada com os dados não-alinhados.

Tabela 6.1 – RMS para cada uma das diferentes arquiteturas treinadas com seqüências sem alinhamento prévio. O valor apresentado na tabela refere-se à melhor época treinada.

Neurônios de Entrada	Neurônios na Camada Oculta	Época com menor RMS	RMS do Treino	RMS do Teste
324	1	120	0,2899	0,3691
324	5	20	0,3518	0,5225
324	10	20	0,4484	0,4999
324	15	25	0,4453	0,4680
324	20	50	0,4201	0,4640
284	1	30	0,4294	0,4923
284	5	20	0,4897	0,4232
284	10	20	0,4797	0,4395
284	15	25	0,4724	0,4383
244	1	30	0,4086	0,5024
244	3	20	0,4308	0,49703
244	5	20	0,4194	0,48210
244	10	40	0,4398	0,4800

Como observado, o RMS obtido – tanto para os conjuntos de treinamento quanto para os conjuntos de teste – foi consideravelmente alto em todas as arquiteturas. Mesmo quando a informação localizada após o TSS foi removida, ou seja, quando os nucleotídeos finais foram excluídos, o RMS não diminuiu. Isto significa que a RN errou exemplos verdadeiros (principalmente seqüências promotoras) e não realiza uma boa predição de promotores (ver resultados de classificação na subseção 6.1.3).

No treinamento da RN com as seqüências alinhadas, os resultados obtidos foram relativamente satisfatórios. Na Tabela 6.2, estão os valores de RMS obtidos

para as diferentes arquiteturas treinadas e na Figura 6.1 está o gráfico dos valores de RMS médio obtido para a arquitetura que apresentou o menor valor.

Tabela 6.2 – Diferentes arquiteturas treinadas para melhor caracterização da metodologia utilizando as seqüências previamente alinhadas.

Neurônios de Entrada	Neurônios na Camada Oculta	Época com menor RMS	RMS do Treino	RMS do Teste
288	1	50	0,2505	0,3382
288	2	30	0,2443	0,3216
288	3	30	0,2301	0,3303
288	4	30	0,2138	0,3245
288	5	30	0,2135	0,3239

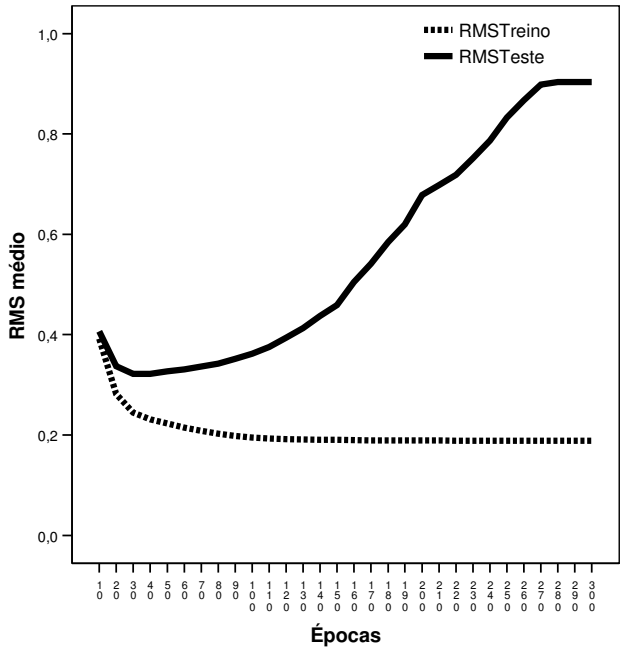

Figura 6.1 – RMS obtido para os conjuntos de treino e de teste para a rede com 288 neurônios na camada de entrada, 2 neurônios na camada oculta e uma saída.

Observa-se que o RMS possui um valor mais baixo para cada arquitetura de rede quando comparado com os dados não alinhados. Assim, observamos que o aprendizado pela RN é mais efetivo com os resultados alinhados porque o alinhamento reduz a falta de sincronismo nos dados de entrada. Após a simulação

com 2 neurônios na camada oculta, o incremento de neurônios nesta camada não melhorou significativamente a predição.

A partir deste conjunto de simulações, obtivemos uma arquitetura de rede que consegue caracterizar e predizer promotores adequadamente. Com bases nestas simulações, a arquitetura escolhida foi com 288 neurônios de entrada (quantidade de pb da seqüência promotora multiplicada pela quantidade de dígitos da codificação ortogonal), 2 neurônios na camada oculta e 1 neurônio na saída, conforme ilustrado a seguir, na figura 6.2:

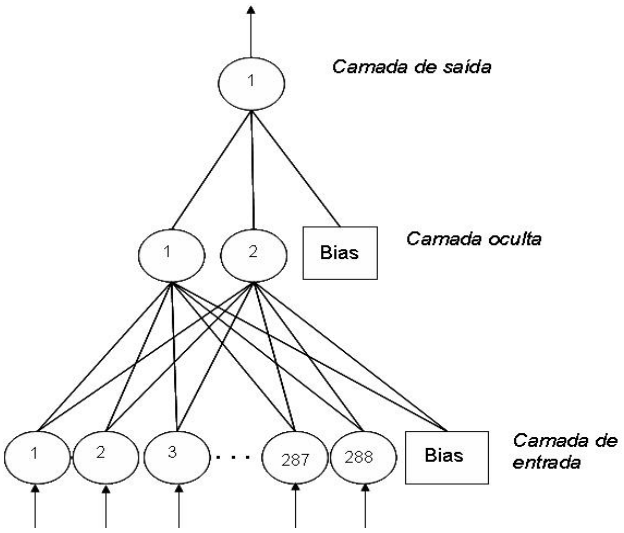

Figura 6.2 – Arquitetura de RN que melhor classificou as seqüências analisadas. Nela, há 288 neurônios de entrada, 2 neurônios na camada escondida e 1 neurônio na camada de saída.

Na próxima subseção, apresenta-se a arquitetura que melhor classificou as seqüências da **Simulação 2**.

6.1.2 Determinação da melhor arquitetura para a Simulação 2

Aqui apresenta-se os resultados obtidos com as redes treinadas com os valores de estabilidade do promotor. Na Tabela 6.3 encontra-se o RMS obtido na melhor época de cada arquitetura treinada com os dados não-suavizados.

Nestas simulações, assim como observado para a **Simulação 1**, que foi treinada com as seqüências não-alinhadas, observamos que o RMS obtido – tanto para os conjuntos de treinamento quanto para os conjuntos de teste – foi consideravelmente alto. Percebemos, então, a importância do sincronismo dos dados de entrada para que a RN consiga realizar um bom aprendizado. No caso da utilização da informação de estabilidade da molécula, não há como sincronizar os dados de entrada, uma vez que o alinhamento insere *gaps,* e não é possível calcular a estabilidade de um *gap*.

Tabela 6.3 – RMS para cada uma das diferentes arquiteturas treinadas com seqüências com os valores de estabilidade não suavizados. O valor apresentado na tabela refere-se à melhor época treinada. O símbolo "–" indica que todas as épocas possuem o mesmo valor de RMS.

Neurônios de Entrada	Número de pb da janela aplicada na fórmula 5.1	Neurônios na Camada Oculta	Época com menor RMS	RMS do Treino	RMS do Teste
80	2	1	50	0,4562	0,4806
80	2	2	50	0,4320	0,4730
80	2	3	40	0,4095	0,4740
80	2	4	40	0,3870	0,4684
80	2	5	30	0,4014	0,4710
80	2	6	40	0,3833	0,4783
80	2	7	30	0,3979	0,4807
80	2	8	30	0,4038	0,4830
79	3	1	50	0,4630	0,4969
79	3	2	60	0,4225	0,4782
79	3	3	50	0,4298	0,4866
79	3	4	60	0,4067	0,4807
79	3	5	60	0,3855	0,4859

79	3	6	30	0,4242	0,4893
79	3	7	20	0,4478	0,4871
79	3	8	20	0,4502	0,4881
78	4	1	60	0,4849	0,4948
78	4	2	80	0,4828	0,4948
78	4	3	80	0,4475	0,4890
78	4	4	30	0,4656	0,492
78	4	5	20	0,478	0,4901
78	4	6	30	0,4513	0,4863
78	4	7	50	0,4320	0,4823
78	4	8	30	0,4534	0,4874
76	6	1	20	0,4950	0,4968
76	6	2	30	0,4875	0,4962
76	6	3	60	0,4592	0,4865
76	6	4	40	0,4758	0,4923
76	6	5	20	0,4803	0,4933
76	6	6	30	0,4711	0,488
76	6	7	30	0,476	0,4931
76	6	8	30	0,4722	0,4911
72	10	1	-	0,5	0,5
72	10	2	90	0,4971	0,4995
72	10	3	80	0,4976	0,4999
72	10	4	40	0,4968	0,4906
72	10	5	30	0,4772	0,4892
72	10	6	30	0,4821	0,4921
72	10	7	40	0,4719	0,4895
72	10	8	40	0,4719	0,4895
68	14	1	-	0,5	0,5
68	14	2	120	0,4923	0,4982
68	14	3	30	0,4944	0,4978
68	14	4	30	0,4946	0,4996
68	14	5	90	0,4675	0,4901
68	14	6	110	0,4754	0,4929
68	14	7	130	0,4705	0,4939
68	14	8	130	0,4722	0,4959
67	15	1	-	0,5	0,5

67	15	2	-	0,4999	0,4999
67	15	3	60	0,4821	0,4992
67	15	4	70	0,4825	0,4967
67	15	5	20	0,4799	0,4975
67	15	6	20	0,4812	0,4967
67	15	7	20	0,4825	0,4967
67	15	8	20	0,4877	0,4983

Como estas redes apresentaram um RMS muito elevado, procuramos uma maneira de diminuir esse valor. Para isso, optou-se pela suavização dos dados de estabilidade. Outra informação relevante sobre os resultados desta simulação é que quanto maior o tamanho da janela empregada na fórmula 5.1, pior é o resultado da classificação (ver subseção 6.1.4). Assim, para a realização da simulação com os valores de estabilidade suavizados, utilizaram-se apenas os valores obtidos com a janela de dois nucleotídeos. A tabela 6.4 apresenta os valores de RMS obtidos para as diferentes arquiteturas treinadas com os valores de estabilidade suavizados.

Tabela 6.4 – RMS para cada uma das diferentes arquiteturas treinadas com seqüências com os valores de estabilidade suavizados. O valor apresentado na tabela refere-se à melhor época treinada.

Neurônios de Entrada	Grau de Suavização	Neurônios na Camada Oculta	Época com menor RMS	RMS do Treino	RMS do Teste
80	3	1	60	0,4548	0,46739
80	3	2	40	0,43811	0,4503
80	3	3	40	0,4284	0,4427
80	3	4	40	0,40378	0,4358
80	3	5	40	0,39805	0,43302
80	3	6	30	0,40375	0,43322
80	3	7	40	0,3966	0,4339
80	3	8	40	0,3985	0,43293
80	5	1	60	0,4663	0,47033
80	5	2	50	0,4175	0,4329
80	5	3	40	0,4097	0,42665

80	5	4	40	0,4040	0,42599
80	5	5	40	0,40579	0,4301
80	5	6	40	0,3972	0,4288
80	5	7	40	0,3972	0,4285
80	5	8	40	0,3995	0,4324
80	7	1	60	0,4471	0,4559
80	7	2	60	0,45263	0,4609
80	7	3	50	0,40479	0,42587
80	7	4	50	0,4022	0,4268
80	7	5	40	0,4125	0,43144
80	7	6	40	0,40877	0,42638
80	7	7	40	0,4471	0,4559
80	7	8	40	0,3993	0,4274
80	9	1	80	0,4686	0,4698
80	9	2	50	0,42141	0,4331
80	9	3	50	0,41088	0,42794
80	9	4	40	0,4057	0,42305
80	9	5	50	0,3998	0,4249
80	9	6	40	0,41029	0,4306
80	9	7	40	0,399	0,4281
80	9	8	40	0,403	0,4248
80	15	1	100	0,4362	0,4436
80	15	2	70	0,4393	0,4470
80	15	3	60	0,4185	0,4327
80	15	4	60	0,4036	0,4207
80	15	5	50	0,3988	0,4224
80	15	6	40	0,40622	0,42399
80	15	7	40	0,40557	0,42178
80	15	8	40	0,40321	0,4219
80	20	1	90	0,4257	0,4336
80	20	2	70	0,4428	0,4503
80	20	3	60	0,4131	0,42631
80	20	4	40	0,4053	0,4193
80	20	5	40	0,4058	0,4194
80	20	6	40	0,4049	0,42014
80	20	7	50	0,4045	0,42503

80	20	8	40	0,4034	0,4227

O gráfico do RMS médio obtido para a melhor arquitetura de RN treinada para esta simulação é visualizado na figura 6.3.

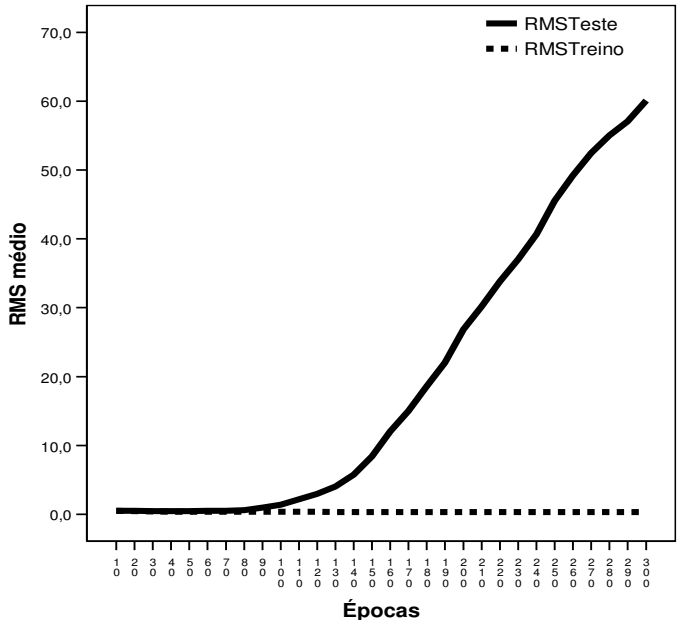

Figura 6.3 – RMS obtido para os conjuntos de treino e de teste para a rede com 80 neurônios na camada de entrada, 4 neurônios na camada oculta e uma saída.

Esta simulação com os dados suavizados apresenta valores menores de RMS quando comparado com os dados não suavizados. Assim, para representar a simulação de predição de promotores com os valores de estabilidade, optou-se pelas simulações que empregaram os valores de estabilidade suavizados. Neste conjunto de simulações, a arquitetura que melhor classificou as seqüências promotoras teve grau 5 de suavização dos valores de estabilidade e 4 neurônios na camada oculta, conforme ilustrado na figura 6.4.

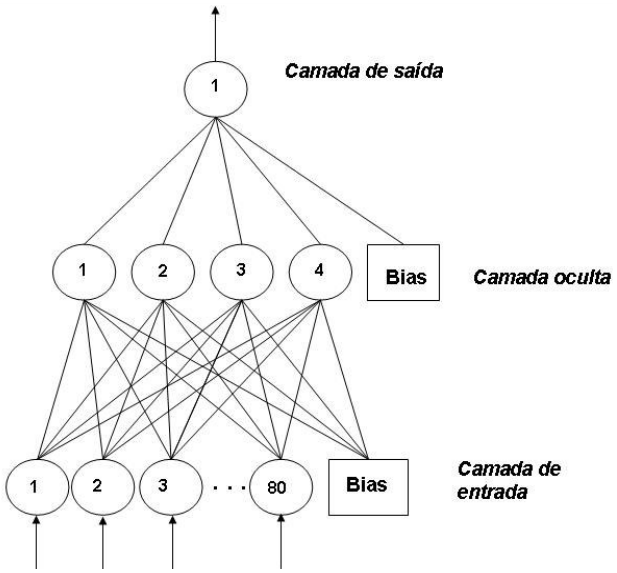

Figura 6.4 – Arquitetura de RN que melhor classificou as seqüências analisadas. Nela, há 80 neurônios de entrada, 4 neurônios na camada escondida e 1 neurônio na camada de saída.

Nas próximas subseções apresenta-se a análise da exatidão média obtida a partir das matrizes de confusão das diferentes arquiteturas treinadas na **Simulação 1** e na **Simulação 2**.

6.1.3 Análise da classificação para a Simulação 1

A análise da classificação considerou os valores de exatidão média e os índices de sensibilidade e especificidade. Estes dados provêm da matriz de confusão obtida para cada um dos *folds* da metodologia 10-*fold cross-validation.* Para melhor análise, ao longo desta subseção, apresenta-se a média das matrizes de confusão obtida para a arquitetura que melhor classificou as seqüências em um dado conjunto de simulação.

Na **Simulação 1** com os dados não alinhados, as matrizes de confusão (ver anexos A, B e C) mostram que a RN falha ao classificar as seqüências promotoras, já que classificou aproximadamente 40% dos promotores como não-promotores.

Apesar de não classificar bem as seqüências promotoras, as seqüências aleatórias, em sua maioria, foram classificadas corretamente. O índice médio de FP apresentado é de aproximadamente 15% para estas simulações. A seguir, a Tabela 6.5 mostra a exatidão média obtida para todas as arquiteturas de redes treinadas com os dados não alinhados. As matrizes de confusão para estas arquiteturas estão nos anexos A, B e C.

Tabela 6.5 – Exatidão média obtida para as arquiteturas treinadas que apresentaram o menor RMS. Estes valores são referentes aos resultados com as seqüências não-alinhadas previamente.

Neurônios de Entrada	Neurônios na Camada Oculta	Época com menor RMS	Exatidão Média	Desvio Padrão
324	1	120	0,71	0,03
324	5	20	0,72	0,02
324	10	20	0,7	0,03
324	15	25	0,74	0,02
324	20	50	0,74	0,02
284	1	30	0,74	0,03
284	5	20	0,7	0,03
284	10	20	0,7	0,02
284	15	25	0,74	0,02
244	1	30	0,7	0,02
244	3	20	0,7	0,02
244	5	20	0,7	0,02
244	10	40	0,72	0,02

Após a análise dos dados não-alinhados, realizou-se a simulação com os dados de entrada alinhados. Como esta simulação apresentou RMS com valores mais baixos, a matriz de confusão, conseqüentemente, mostra índices de FP e FN mais baixos. Na Tabela 6.6, apresenta-se a média das matrizes de confusão para as simulações da arquitetura que melhor classificou as seqüências alinhadas: 288 neurônios na camada de entrada, 2 neurônios na camada oculta e 1 neurônio na

camada de saída (conforme descrita na subsecção 6.1.1). As matrizes de confusão para as demais arquiteturas encontram-se no anexo D.

Tabela 6.6 – Matriz de confusão média para a arquitetura com 288 neurônios na camada de entrada, 2 neurônios na camada oculta e 1 neurônio na camada de saída.

	Classificado como promotor	Classificado como não-promotor
Promotor	70	24
Não promotor	9	85

Aqui, podemos observar que a RN consegue classificar os promotores com maior precisão, já que o índice de seqüências promotoras classificadas como não promotoras diminuiu - aproximadamente 23% dos promotores foram classificados como não-promotores. Além da melhora na classificação dos promotores, a média do índice de FP foi de aproximadamente 10%. Assim, quando comparados os resultados obtidos com as seqüências não alinhadas e alinhadas, vê-se que o alinhamento é de fundamental importância para que os índices de exatidão da RN sejam satisfatórios.

Para a completa análise da exatidão da melhor arquitetura que classificou os promotores previamente alinhados, a Figura 6.5 mostra a curva ROC obtida com os índices de especificidade e sensibilidade. Além disso, vê-se refletida na curva que a RN classifica acertadamente a maioria das seqüências aleatórias, ou seja, possui maior especificidade do que sensibilidade.

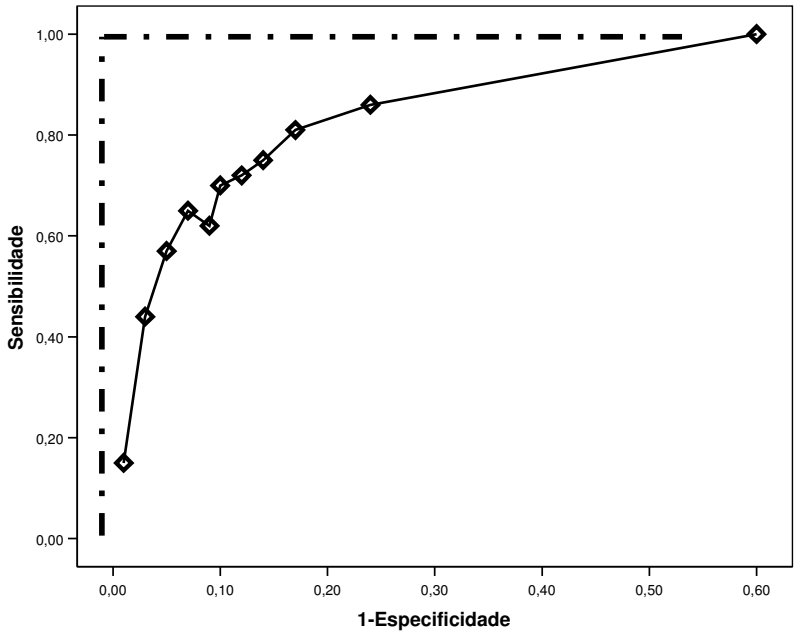

Figura 6.5 – Curva ROC para a arquitetura de RN que melhor classificou as seqüências promotoras previamente alinhadas. No canto esquerdo (pontilhado), a curva ROC ideal.

O incremento de neurônios na camada oculta não aumentou significativamente os índices considerados na análise da classificação, conforme pode ser visto na matriz de confusão da Tabela 6.7, obtida a partir de uma arquitetura mais complexa que a escolhida neste trabalho. Os valores de exatidão média para todas as arquiteturas treinadas com os promotores previamente alinhados (Tabela 6.8) justificam a escolha pela arquitetura mais simples.

Tabela 6.7 – Matriz de confusão média para a arquitetura com 288 neurônios na camada de entrada, 4 neurônios na camada oculta e 1 neurônio na camada de saída.

	Classificado como promotor	Classificado como não-promotor
Promotor	71	23
Não promotor	18	76

Tabela 6.8 – Exatidão média obtida para as arquiteturas treinadas que apresentaram o menor RMS. Estes valores são referentes aos resultados com as seqüências previamente alinhadas.

Número de Entradas	Número de Neurônios na Camada Oculta	Época com menor RMS	Exatidão Média	Desvio Padrão
288	1	50	0,76	0,04
288	2	30	0,8	0,04
288	3	30	0,8	0,03
288	4	30	0,81	0,023
288	5	30	0,81	0,025

Na próxima seção, estão os resultados de classificação obtida para a **Simulação 2**.

6.1.4 Análise da classificação para a Simulação 2

Para a análise da classificação desta simulação, que empregou os valores de estabilidade da seqüência, também se considerou os valores de exatidão média e os índices de sensibilidade e especificidade provenientes das matrizes de confusão obtidas para cada um dos *folds* da metodologia 10-*fold cross-validation.*

Para o conjunto de simulações com os dados não suavizados, a exatidão média obtida para as arquiteturas treinadas está na Tabela 6.9 e as matrizes de confusão para estas arquiteturas estão apresentadas nos anexos E, F, G e H. Os resultados obtidos com as arquiteturas treinadas com as seqüências resultantes da aplicação da janela de 10 ou mais pb na equação 5.1, não são apresentados porque estas redes não conseguiram índices de RMS satisfatórios e o valor de saída para os exemplos de entrada destas arquiteturas era sempre o mesmo: 0,5. Assim, como a rede não conseguiu aprender com estes dados, estas simulações não foram consideradas.

Tabela 6.9 – Exatidão média para cada uma das diferentes arquiteturas treinadas com seqüências com os valores de estabilidade não suavizados.

Neurônios de Entrada	Número de pb da janela aplicada na fórmula 6.1	Neurônios na Camada Oculta	Época com menor RMS	Exatidão Média	Desvio Padrão
80	2	1	50	0,61	0,05
80	2	2	50	0,64	0,03
80	2	3	40	0,68	0,02
80	2	4	40	0,68	0,02
80	2	5	30	0,68	0,02
80	2	6	40	0,68	0,02
80	2	7	30	0,66	0,01
80	2	8	30	0,66	0,02
79	3	1	50	0,57	0,06
79	3	2	60	0,63	0,01
79	3	3	50	0,63	0,04
79	3	4	60	0,65	0,02
79	3	5	60	0,66	0,02
79	3	6	30	0,63	0,02
79	3	7	20	0,62	0,02
79	3	8	20	0,62	0,02
78	4	1	60	0,57	0,11
78	4	2	80	0,56	0,06
78	4	3	80	0,6	0,02
78	4	4	30	0,6	0,02
78	4	5	20	0,59	0,02
78	4	6	30	0,62	0,05
78	4	7	50	0,63	0,02
78	4	8	30	0,62	0,02
76	6	1	20	0,51	0,01
76	6	2	30	0,52	0,01
76	6	3	60	0,6	0,03
76	6	4	40	0,58	0,01
76	6	5	20	0,61	0,05
76	6	6	30	0,62	0,03
76	6	7	30	0,59	0,02
76	6	8	30	0,59	0,02

Nestas simulações, observamos que a exatidão média de todas as arquiteturas fica próximo aos 60% de acerto. Os valores maiores que este são obtidos somente com os dados de estabilidade provenientes da aplicação da fórmula com 2 pb. Isto se deve ao fato de que, quanto maior a janela, menor é a quantidade de valores, sendo suprimida a informação referente aos nucleotídeos após o TSS. Sabe-se que a estabilidade é importante quando comparamos os seus valores antes e depois do TSS. Assim, percebe-se que a RN necessita, nesta simulação, da seqüência promotora em sua totalidade.

Após a análise destes resultados, optou-se por realizar simulações com os valores de estabilidade suavizados para melhorar os índices de exatidão. A Tabela 6.10 apresenta a exatidão média obtida para todas as arquiteturas de redes treinadas com os dados suavizados.

Tabela 6.10 – Exatidão média para cada uma das diferentes arquiteturas treinadas com seqüências com os valores de estabilidade suavizados.

Neurônios Entrada	Grau de Suavização	Neurônios na Camada Oculta	Época com menor RMS	Exatidão Média	Desvio Padrão
80	3	1	60	0,63	0,04
80	3	2	40	0,67	0,02
80	3	3	40	0,7	0,02
80	3	4	40	0,73	0,02
80	3	5	40	0,73	0,02
80	3	6	30	0,73	0,02
80	3	7	40	0,73	0,02
80	3	8	40	0,73	0,02
80	5	1	60	0,62	0,09
80	5	2	50	0,72	0,03
80	5	3	40	0,72	0,02
80	5	4	40	0,74	0,02
80	5	5	40	0,73	0,02
80	5	6	40	0,74	0,02

80	5	7	40	0,74	0,02
80	5	8	40	0,73	0,02
80	7	1	60	0,65	0,05
80	7	2	60	0,64	0,06
80	7	3	50	0,72	0,03
80	7	4	50	0,73	0,02
80	7	5	40	0,72	0,03
80	7	6	40	0,73	0,03
80	7	7	40	0,74	0,02
80	7	8	40	0,73	0,02
80	9	1	80	0,61	0,07
80	9	2	50	0,7	0,03
80	9	3	50	0,72	0,03
80	9	4	40	0,74	0,02
80	9	5	50	0,74	0,02
80	9	6	40	0,72	0,02
80	9	7	40	0,74	0,03
80	9	8	40	0,74	0,03
80	15	1	100	0,68	0,04
80	15	2	70	0,67	0,03
80	15	3	60	0,71	0,03
80	15	4	60	0,74	0,02
80	15	5	50	0,74	0,03
80	15	6	40	0,74	0,02
80	15	7	40	0,74	0,03
80	15	8	40	0,74	0,02
80	20	1	90	0,7	0,02
80	20	2	70	0,66	0,04
80	20	3	60	0,71	0,03
80	20	4	40	0,74	0,02
80	20	5	40	0,74	0,02
80	20	6	40	0,74	0,02
80	20	7	50	0,73	0,03
80	20	8	40	0,74	0,02

Como pode ser visto na Tabela 6.10 a arquitetura mais simples com maior exatidão foi com grau de suavização 5 e 4 neurônios na camada oculta. Apesar de que a mesma arquitetura com os dados suavizados em grau 3, 7 e 9 apresentarem um valor de exatidão próximo ao conseguido pela arquitetura escolhida no trabalho, as suas matizes de confusão apresentadas nos anexos I, J, L, M,N e O, justificam sua exclusão como resultado ótimo. Nelas, os resultados da classificação das seqüências promotoras são inferiores ao obtido com a rede escolhida neste trabalho.

A média das matrizes de confusão para as simulações da arquitetura que melhor classificou os valores suavizados está na Tabela 6.11, a seguir.

Tabela 6.11 – Matriz de confusão média para a arquitetura com 80 neurônios na camada de entrada, 4 neurônios na camada oculta e 1 neurônio na camada de saída. O grau de suavização destes dados foi 5.

	Classificado como promotor	Classificado como não-promotor
Promotor	68	26
Não promotor	21	73

Esta matriz de confusão nos mostra que a RN tem mais dificuldade em classificar corretamente tanto as seqüências aleatórias quanto as seqüências promotoras. A curva ROC obtida com os índices de especificidade e sensibilidade para esta matriz pode ser vista na Figura 6.6. Nela, vemos que a curva se distancia do canto superior esquerdo, mostrando que para esta simulação, é mais difícil conseguir índices satisfatórios de verdadeiros positivos com verdadeiros negativos (resultado já observado na matriz de confusão).

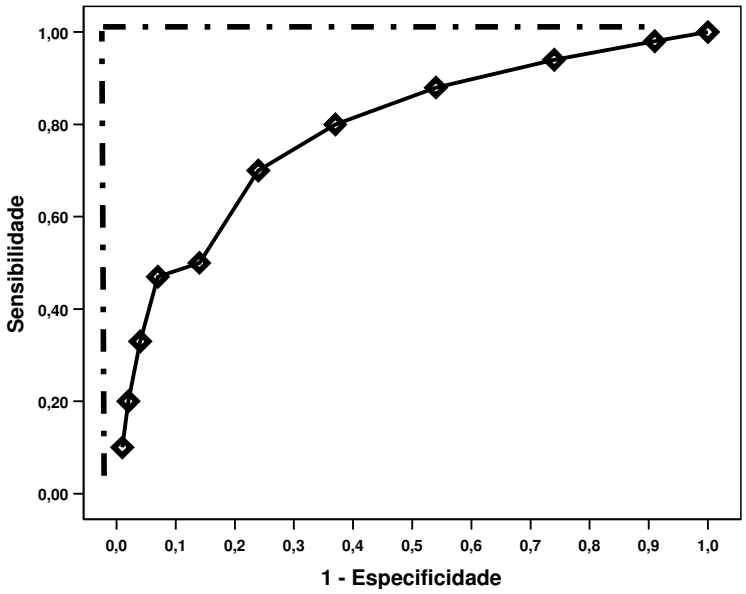

Figura 6.6 – Curva ROC para a arquitetura de RN que melhor classificou as seqüências promotoras com valores de estabilidade suavizados. No canto esquerdo (pontilhado), a curva ROC ideal.

A próxima seção apresenta as regras extraídas e suas respectivas análises.

6.2 REGRAS EXTRAÍDAS A PARTIR DAS REDES NEURAIS TREINADAS

Nesta seção, estão descritas as regras extraídas a partir dos pesos dos neurônios ocultos da RN treinada e também da árvore de decisão. Na primeira subseção, estão as regras extraídas para a melhor arquitetura da **Simulação 1**, enquanto que na segunda subseção estão as regras obtidas para a **Simulação 2**.

6.2.1 Regras extraídas para a melhor arquitetura da Simulação 1

Utilizando os pesos dos neurônios da camada oculta da RN, foram extraídas 5 regras. Destas, 3 regras geradas classificaram as seqüências como não-promotoras, totalizando 1124 seqüências. De todas estas seqüências, apenas 335 delas eram promotoras, ou seja, foram classificadas erroneamente pela rede. Uma

regra não classificou as seqüências, pois a RN atribui o valor de 0,5 para as seqüências. Esta regra incluiu 277 seqüências, sendo que 196 delas promotoras.

A regra que classificou as seqüências como promotoras incluiu 479 seqüências. Aqui, constavam apenas 70 seqüências aleatórias. Como esta regra classifica as seqüências como promotoras, seu protótipo foi analisado. Consideramos como protótipo a seqüência com os nucleotídeos que recebem maior peso durante o aprendizado (conforme metodologia descrita na subseção 4.4.2). O protótipo é considerado como o padrão de seqüência mais observado em todos os promotores analisados. O protótipo para esta regra foi:

SE Promotor *ENTÃO*

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
T	(T|A)	T	A	T	(A|T)	A	A	A	(A|T)	(A|T)	T	T	T	T	(A|T|G)	(A|T)

18	19	20	21	22	23	24	25	26	27	28	29	30	31
(T|A)	(T|A)	A	A	(T|A)	(A|T)	T	**T**	T	(A|T|C)	A	A	A	(T|G)

32	33	34	35	36	37	38	39	40	41	42	43	44	45
T	T	(A|T)	A	T	T	(A|T|C)	A|C	T	A	T	(A|T|C)	A	**T**

46	47	48	49	50	51	52	53	54	55	56	57	58	59
A	**T**	**T**	T	T	T	A	A	T	T	(A|T)	A	T	(A|T)

60	61	62	63	64	65	66	67	68	69	70	71	72
(T|A)	(T|G)	(T|A)	(T|A)	(A|T)	A	T	A	A	T	(T|G)	(A|T|C)	(A|T)

Nesta regra, todos os nucleotídeos que estão em sublinhados são os que possuem o maior papel no aprendizado. Estes nucleotídeos são considerados como determinantes no aprendizado porque possuem maior valor no protótipo obtido na regra (valor superior a 0,4). Os nucleotídeos entre parênteses indicam que ambos possuem importância equivalente no aprendizado (o símbolo | significa *ou*), já que possuem coeficientes com valores muito próximos. As regiões biologicamente

importantes estão localizadas, aproximadamente, nas posições 22 até 26 (região -35) e entre os nucleotídeos 44 e 51 (região -10). Assim, vemos em negrito o nucleotídeo central da região -35 coincidindo com o padrão biológico. Já na região -10 é possível ver 4 nucleotídeos importantes para o aprendizado da RN que coincidem com o padrão biológico (em negrito). Além dos hexâmeros sabidamente característicos dos promotores, a rede também usa outros nucleotídeos para realizar a classificação, sendo que não há função biológica descrita na literatura para eles.

Estes padrões, também são observáveis na árvore de decisão obtida com o software WEKA (ver Figura 6.7). Para esta árvore, utilizou-se 73 atributos (72 nucleotídeos e o indicador se a seqüência é ou não promotora) e 1880 seqüências, sendo 940 promotores previamente alinhados e 940 aleatórias. A metodologia de 10-*fold-cross-validation* foi a escolhida para a geração de resultados estatisticamente válidos.

```
nucleotideo_25 = A
|   nucleotideo_45 = A: NP (72.0/18.0)
|   nucleotideo_45 = T
|   |   nucleotideo_46 = A: P (45.0/18.0)
|   |   nucleotideo_46 = T: NP (22.0/9.0)
|   |   nucleotideo_46 = C: NP (23.0/3.0)
|   |   nucleotideo_46 = G: P (41.0/15.0)
|   nucleotideo_45 = C: NP (75.0/14.0)
|   nucleotideo_45 = G: NP (68.0/7.0)
nucleotideo_25 = T
|   nucleotideo_47 = A: P (125.0/54.0)
|   nucleotideo_47 = T: P (324.0/58.0)
|   nucleotideo_47 = C
|   |   nucleotideo_50 = A: P (45.0/10.0)
|   |   nucleotideo_50 = T: P (42.0/8.0)
|   |   nucleotideo_50 = C: NP (33.0/13.0)
|   |   nucleotideo_50 = G: NP (31.0/11.0)
|   nucleotideo_47 = G: NP (112.0/44.0)
nucleotideo_25 = C
|   nucleotideo_45 = A: NP (90.0/28.0)
|   nucleotideo_45 = T: P (222.0/64.0)
|   nucleotideo_45 = C: NP (102.0/42.0)
|   nucleotideo_45 = G: NP (95.0/34.0)
nucleotideo_25 = G: NP (309.0/97.0)
```

Figura 6.7 – Árvore de decisão gerada pelo algoritmo J-48 do software WEKA com as seqüências promotoras previamente alinhadas.

A árvore gerada possui 25 folhas, sendo que 67% dos exemplos foram classificados corretamente e a precisão para as seqüências promotoras foi de 68%. O nucleotídeo raiz desta árvore é o localizado na posição 25, que está em uma posição biologicamente importante: a região -35. Este nucleotídeo também foi importante para o aprendizado da RN, como vimos na regra obtida com o peso dos neurônios da camada oculta. A presença do nucleotídeo G nesta posição é suficiente para classificar a seqüência como não-promotora. Esta regra tem fundamentação biológica, já que no clássico trabalho de Lisser e Margalit (1993), poucas são as seqüências promotoras que possuem um G nesta posição. Outra informação relevante observável a partir desta árvore é que todos os nucleotídeos dela estão presentes nas duas regiões biologicamente importantes. Algumas regras que podem ser observadas na árvore são discutidas a seguir, onde **n** é a abreviação de nucleotídeo.

- SE Promotor ENTÃO (n_25 = A e n_45 = T e n_46 = A ou G)

- SE Promotor ENTÃO (n_25 = T e n_47 = A ou T)

- SE Promotor ENTÃO (n_25 = T e n_47 = C e n_50 = A ou T)

- SE Promotor ENTÃO (n_25 = C e n_45 = T)

As regras mostram a relação entre os nucleotídeos localizados na região -10 e na região -35. Isto é de relevância biológica, já que a RNAp, ao ligar-se na região promotora, deve estabelecer ligações com determinados nucleotídeos das duas regiões ao mesmo tempo (devido ao seu tamanho). Assim, não considerar a correlação entre o conteúdo de determinada posição com outra não contribui para o

pleno entendimento de como ocorre a expressão gênica ou para a caracterização das seqüências promotoras.

Comparando-se os resultados das duas metodologias de extração de regras, vemos a importância de determinados nucleotídeos em posições com função biológica descrita na classificação da seqüência como promotora. As duas abordagens de extração de regras mostram a importância da T nas posições 25 e 47. Além disso, outras posições também aparecem com importância para a caracterização da seqüência, mas sem alguma função biológica descrita. Estes podem estar relacionados com a ligação entre a RNAp e o promotor.

6.2.2 Regras extraídas para a melhor arquitetura da Simulação 2

A partir da metodologia que utiliza os pesos dos neurônios da camada oculta da RN, foram extraídas 9 regras, conforme mostrado na Figura 6.8. Para melhor compreensão destas regras, apresentamos o protótipo como um gráfico e não na forma de escrita formal, já que as seqüências são valores numéricos.

Destas regras, apenas 2 classificaram as seqüências como não-promotoras (regras 1 e 2 da figura 6.8). O total de seqüências desta regra foi 1273, sendo 449 promotores e as restantes seqüências aleatórias. As demais regras geradas classificaram as seqüências como promotoras.

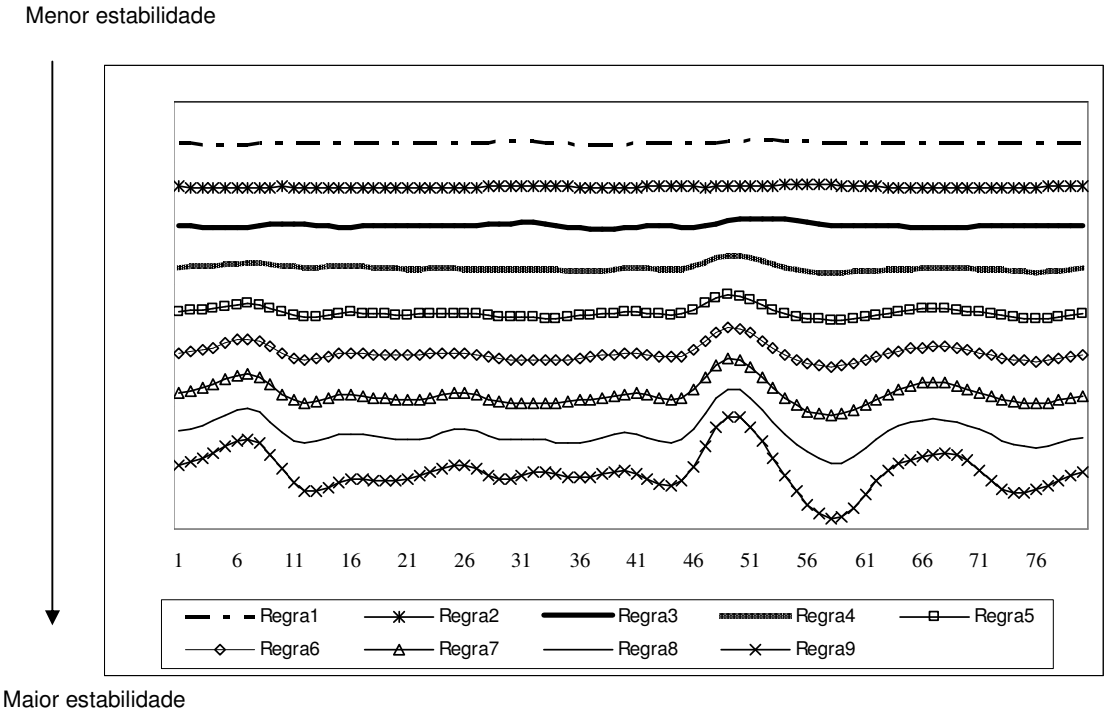

Figura 6.8 – Regras geradas para os valores de estabilidade das seqüências promotoras. Aqui, a primeira linha corresponde à regra 1, seguindo em ordem crescente o número das demais regras.

Para as regras que classificaram as seqüências promotoras (regras 3 até 9), vemos que 6 delas (as regras de 4 a 9) caracterizam seqüências que apresentam uma diminuição dos valores de estabilidade próximo à região do nucleotídeo 49 (região -10). Estas regras, apesar de mostrarem uma informação biologicamente coerente com a literatura sobre os promotores, compreendem a minoria das seqüências do treinamento: 271 exemplos. Destas, 105 eram seqüências não promotoras. Já a regra que compreende o maior número de seqüências: a regra 3, não mostra o declínio de estabilidade próximo ao nucleotídeo 49 tão evidente quanto às demais regras. Somente nela, foram incluídas 347 seqüências, sendo que 325 destas eram promotoras. Este resultado explica a fraca classificação de seqüências promotoras nesta simulação.

A árvore de decisão obtida com estes dados (ver Figura 6.9) teve 11 folhas e um 67% dos exemplos foram corretamente classificados. O índice de FP obtidos para os promotores foi de 30%.

```
nucleotideo_49 <= -1
|   nucleotideo_51 <= -1.28
|   |   nucleotideo_55 <= -1.28: NP (608.0/149.0)
|   |   nucleotideo_55 > -1.28
|   |   |   nucleotideo_27 <= -1.45: NP (105.0/32.0)
|   |   |   nucleotideo_27 > -1.45
|   |   |   |   nucleotideo_17 <= -1.3: NP (92.0/38.0)
|   |   |   |   nucleotideo_17 > -1.3: P (92.0/37.0)
|   nucleotideo_51 > -1.28
|   |   nucleotideo_49 <= -1.84: NP (108.0/45.0)
|   |   nucleotideo_49 > -1.84
|   |   |   nucleotideo_32 <= -1.3
|   |   |   |   nucleotideo_16 <= -1.3: NP (100.0/41.0)
|   |   |   |   nucleotideo_16 > -1.3: P (86.0/26.0)
|   |   |   nucleotideo_32 > -1.3: P (189.0/43.0)
nucleotideo_49 > -1
|   nucleotideo_53 <= -1.3
|   |   nucleotideo_7 <= -1.28: NP (121.0/57.0)
|   |   nucleotideo_7 > -1.28: P (84.0/21.0)
|   nucleotideo_53 > -1.3: P (296.0/41.0)
```

Figura 6.9 – Árvore de decisão gerada pelo algoritmo J-48 do software WEKA com os valores de estabilidade das seqüências promotoras.

Algumas regras que podem ser vistas na árvore de decisão acima são discutidas a seguir, onde **n** é a abreviação de nucleotídeo.

- SE Promotor ENTÃO n_49 <= -1 e n_51 <= -1,28 e n_55 > -1,28 e n_27 > -1,45 e n_17 > -1,3

- SE Promotor ENTÃO n_49 <= -1 e n_51> -1,28 e n_49 > -1,84 e n_32 > -1,3

- SE Promotor ENTÃO n_49 <= -1 e n_51> -1,28 e n_49 > -1,84 e n_32 > -1,3 e n_16 > -1,3

- SE Promotor ENTÃO n_49 > -1 e n_53 > -1,3

- SE Promotor ENTÃO n_49 > -1 e n_53 < -1,3 e n_7 > -1,28

Nesta árvore, o nucleotídeo 49 aparece como raiz. Isto é biologicamente explicado já que esta região é a que apresenta menor estabilidade, devido à formação do RP_o no processo de transcrição. Outros nucleotídeos localizados nas regiões biologicamente funcionais (nucleotídeos 51 e 32) também estão presentes nestas regras. Percebe-se, também, a presença de outros nucleotídeos sem importância biológica descrita.

Assim, com a extração de regras, percebemos que apesar da degeneração do padrão biológico ideal, este é um fator importante na caracterização e classificação de uma dada seqüência como promotora. A estabilidade do promotor é um parâmetro que distingue as regiões promotoras de não-promotoras. Apesar da RN treinada com os valores de estabilidade não demonstrar alta acurácia, esta informação auxilia na caracterização das seqüências. As regras também nos auxiliam na compreensão de como a RN aprende a reconhecer as seqüências promotoras e da dificuldade que é para que este aprendizado seja realizado com uma alta taxa de acurácia.

7 CONSIDERAÇÕES FINAIS

A metodologia de RNs mostrou-se adequada para o problema de predição de promotores, já que não foi necessária uma arquitetura complexa para realizar a predição. A exatidão média obtida na melhor arquitetura utilizando a codificação ortogonal da seqüência promotora foi de 80% com desvio padrão de 0,02. Este resultado é explicado pela capacidade que a RN possui no reconhecimento de padrões degenerados. Sabemos que isto é importante porque o consenso estabelecido para os promotores procarióticos é pequeno, degenerado e restrito a um pequeno número de seqüências. Devido a estas características das seqüências, o alinhamento prévio é necessário, pois a rede necessita do sincronismo entre os promotores para realizar uma boa classificação. Apesar da degeneração dos motivos consensuais dos promotores, a extração de regras mostrou que as duas regiões biológicas (-10 e -35) são determinantes para a classificação de uma determinada seqüência como promotora.

Visando a caracterização da seqüência promotora, realizou-se o treinamento com os valores de estabilidade dos promotores. Nesta simulação, a melhor arquitetura obteve exatidão média de 74%, com desvio padrão de 0,02. Este resultado é inferior ao obtido com a codificação ortogonal, mas devemos considerar que, mesmo inferior, este resultado foi conseguido sem o alinhamento prévio dos

dados. Para tentar diminuir a falta de sincronismo, estes dados foram suavizados. No entanto, este procedimento não foi suficiente para que a RN conseguisse uma exatidão alta. As regras para este conjunto de simulações destacam o nucleotídeo da posição 49 como caracterizador da seqüência promotora, já que nesta posição os valores de estabilidade caem. Conforme a literatura e os conhecimentos biológicos já estabelecidos, a região -10 é o local onde a dupla fita de DNA se abre para iniciar a formação do RNAm (processo de transcrição) portanto, explica o fato do nucleotídeo 49 ter destaque nestas simulações.

Assim, a metodologia de RN com codificação ortogonal deste trabalho mostrou uma exatidão média próxima à conseguida por outras metodologias descritas na revisão bibliográfica desta dissertação. Apesar de ser uma importante característica dos promotores, a informação de estabilidade não melhorou a predição. Existem problemas que são inerentes à metodologia empregada e que não possibilitam a melhora na acurácia. Uma delas, amplamente discutida, é a falta de conservação dos motivos consensuais. Outra é o caráter ruidoso dos dados biológicos, evidenciados pelas poucas épocas necessárias para o aprendizado.

Apesar da menor exatidão conseguida com a simulação utilizando a estabilidade acreditamos que, se superada a falta de sincronismo, esta pode ser um bom indicador para o aumento da exatidão no problema de predição de promotores. Uma das alternativas para que os dados de estabilidades possuam maior sincronismo seria a aplicação de *wavelets*. No entanto, este tipo de transformada só pode ser aplicado em seqüências com muitos pontos (mínimo requerido em torno de 500 pontos). Assim, esta metodologia é inaplicável nas seqüências obtidas no banco de dados RegulonDB, que possui apenas 81 pb. Mesmo com uma exatidão menor, a **Simulação 2** é relevante pois contribui para a compreensão biológica dos

promotores, já que as regras mostram mais de um local com queda de estabilidade. Análises experimentais da ligação RNAp-promotor aliadas com trabalhos de IA podem oferecer um caminho para a descoberta do conhecimento relacionado aos promotores e, conseqüentemente, à expressão gênica.

8 REFERÊNCIAS BIBLIOGRÁFICAS

ANDREWS, Robert; DIEDERICH, Joachim; TICKLE, Alan B. A survey and critique of techniques for extracting rules from trained artificial neural networks. **Knowledge-Based Systems***,* v. 8, n. 6, p. 373-389, 1995.

ARBATLI, A. D.; AKIN, H.L. Rule extraction from trained neural networks using genetic algorithms. **Non linear analysis, Theory, Methods e Applications,** v. 30, n. 3, p. 1639-1648, 1997.

BALDI, Pierre; BRUNAK, Soren. **Bioinformatics :** the machine learning approach. 2. ed. Cambridge: MIT, 2001. 351 p.

BARRERA, Junior; CESAR-Jr, Roberto M.; FERREIRA, João E., GUBITOSO, Marco E. An environment for knowledge discovery in biology. **Computers in Biology and Medicine,** v. 34, n.5, p. 427-447, 2004.

BORUKHOV, Sergei; LEE, Joking. RNA polymerase structure and function at *lac* operon. **Computes Rendus Biologies**, v.328, n.6, p. 576-587, 2005.

BURDEN, S.; LIN, Y.-X.; ZHANG, R. Improving promoter prediction for the NNPP2.2 algorithm: a case study using *Escherichia coli* DNA sequences. **Bioinformatics**, v.21, n. 05, p. 601-607, 2005.

BURGESS, Richard R.; ANTHONY, Larry. How sigma docks to RNA polymerase and what sigma does. **Current Opinion in Microbiology**, v. 04, n. 02, p. 126-131, 2001.

CECHIN, Adelmo Luis. **The extraction of Fuzzy Rules from Neural Networks.** 1998. 149 f. Tese (Doutorado em Informática) – Aachen: Shaker, [1998].

CLOETE, Ian.; ZURADA, Jacek M. **Knowledge-Based neurocomputing**. Cambridge: MIT, 2000. 486 p.

COTIK, V.; ZALIZ, R. Romero; ZWIR, I. A hybrid promoter analysis methodology for prokaryotic genomes. **Fuzzy Sets and Systems**, v.152, n. 1, p. 83-102, 2005.

DE ROBERTIS, Eduardo D. P. **Bases da biologia celular e molecular.** 2. ed. Rio de Janeiro: Guanabara Koogan, 1993. 307 p.

DEMELER, Borries; ZHOU, Guangwen. Neural network optimization for *E. coli* promoter prediction. **Nucleic Acids Research**, v. 19, n. 07, p. 1593-1599, 1991.

ESKIN, Eleazar; KEICH, Uri; GELFAND, Mikhail S.; PEVZNER, Pavel A. Genome-wide analysis of bacterial promoter regions. In: **Pacific Symposium in Biocomputing**, p. 29-40, 2003.

GORDON, Leo; CHERVONENKIS, Alexey Ya.; GAMMERMAN, Alex J.; SHAHMURADOV, Ilham A.; SOLOVYEV, Victor V. Sequence alignment for recognition of promoter regions. **Bioinformatics**, v.19, n. 15, p. 1964-1971, 2003.

HATFIELD, G.W.; HUNG, She-Pin; BALDI, Pierre. Differential analysis of DNA microarray gene expression data. **Molecular Microbiology**, v.47, n. 04, p. 871-877, 2003.

HAYKIN, Simon. **Neural networks:** a comprehensive foundation. 2. ed. New Jersey: Prentice-Hall, 1999. 842 p.

HEGGER R., KANTZ H., SCHREIBER T. Practical implementation of nonlinear time series methods: The TISEAN package, **Chaos**, v.2, n. 9, p. 413 – 435, 1999.

HERTZ, Gerald Z.; STORMO, Gary D. Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. **Bioinformatics**, v.15, n. 07/08, p. 563-577, 1999.

HORNIK, Kurt. Multilayer feedforward networks are universal aproximators. **Neural Networks**, v.02, n.5, p. 359-366, 1989.

HOWARD, Daniel; BENSON, Karl. Evolutionary computation method for pattern recognition of *cis*-acting sites. **BioSystems**, v. 72, n.1/2 , p. 19-27, 2002.

HUANG, Samuel H.; XING, Hao. Extract intelligible and concise fuzzy rules from neural networks. **Fuzzy Sets and Systems**, v.132, p. 233-243, 2005

JÁUREGUI, Ruy; ABREU-GOODGER, Cei; MORENO-HAGELSIEB, Gabriel, COLLADO-VIDES, Julio; MERINO, Enrique. Conservation of DNA curvature signals in regulatory regions of prokaryotic genes. **Nucleic Acids Research**, v. 31, n. 23, p. 6770-6777, 2003.

KALATE, Rupali N.; TAMBE, Sanjeev S.; KULKARNI, Bhaskar D. Artificial neural networks for prediction of mycobacterial promoter sequences. **Computational Biology and Chemistry**, v. 27, n. 6, p. 555-564, 2003.

KANHERE, Aditi; BANSAL, Manju. Structural properties of promoters: similarities and differences between prokaryotes and eukaryotes. **Nucleic Acids Research**, v. 33, n. 10, p. 3165-3175, 2005a.

KANHERE, Aditi; BANSAL, Manju. A novel method for prokaryotic promoter prediction based on DNA stability. **BMC Bioinformatics**, v. 6, 2005b. Disponível em http://www.biomedcentral.com/1471-2105/6/1, último acesso em 03 de maio de 2006.

KESELER, Ingrid M.; COLLADO-VIDES, Julio; GAMA-CASTRO, Socorro; IGRAHAM, John; PALEY, Suzanne; PAULSEN, Ian T.; PERALTA-GIL, Martín; KARP, Peter D. EcoCyc: a comprehensive database resource for *Escherichia coli*. **Nucleic Acids Research**, v. 33, p. 334-337, 2005.

KIRYU, Hisanori; OSHIMA, Taku; KIYOSHI, Asai. Extracting relations between promoter sequences and their strengths from microarray data. **Bioinformatics**, v.21, n. 07, p. 1062-1068, 2005.

LEHNINGER, Albert L.; COX, Michael M.; NELSON, David L. **Principles of biochemistry.** 3. ed. New York: Worth, 2000. 1152 p.

LEWIN, Benjamin. **Genes vii.** Porto Alegre: Artes Médicas, 2001. 955 p.

LISSER, Shlomit; MARGALIT, Hanah. Compilation of *E. coli* mRNA promoter sequences. **Nucleic Acids Research**, v. 21, n. 07, p. 1507-1516, 1993.

MAHADEVAN, I.; GHOSH, I. Analysis of *E. coli* promoter structures using neural networks. **Nucleic Acids Research**, v. 22, n. 11, p. 2158-2165, 1994.

MITCHELL, Tom Michael. **Machine Learning**. Boston: WCB/McGraw-Hill, 1997. 414 p.

MOUNT, David W. **Bioinformatics :** sequence and genome analysis. New York: Cold Spring Harbor Laboratory, 2000. 564 p.

MURAKAMI, K. S.; MASUDA, S.; CAMPBELL, E. A.; MUZZIN, O.; DARST, S. A. Structural Basis of Transcription Initiation: An RNA polymerase holoenzyme-DNA complex. **Science**, v.296, n. 5571, p. 1285-1290, 2002.

NARYSHKIN, N.; REVYAKIN A.; KIM, Y. MEKLER, V. EBRIGHT, R. H. Structural organization of the RNA polymerase-promoter open complex. **Cell**, v.101, N. 6, p. 601-611, 2000.

OGATA, Hiroyuki; GOTO, Susumu; SATO, Kazushige; FUJIBUCHI, Wataru; BONO, Hidemasa; KANEHISA, Minoru. KEGG: Kyoto Encyclopedia of Genes and Genomes. **Nucleic Acids Research**, v. 27, n. 1, p. 29-34, 1999.

O'NEILL, Michael C. Training back-propagation neural networks to define and detect DNA-binding sites. **Nucleic Acids Research**, v. 19, n. 2, p. 313-318, 1991.

OPPON, Ekow CruickShank. **Synergistic Use of Promoter Prediction Algorithms: a choice for a small training dataset?.** 2000. 238 f. Tese (Doutorado em Ciência da Computação) – South African National Bioinformatics Institute (SANBI), [2000].

PEDERSEN, A. G.; ENGELBRECHT, J. Investigations of *Escherichia coli* promoter sequences with artificial neural networks: New signals discovered upstream of the transcriptional start point. In: **Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology (ISMB-95)**, v.3, p. 292-299, 1995.

PEDERSEN, A. G.; BALDI, P.; BRUNAK, S.; CHAUVIN, Y. Characterization of prokaryotic and eukaryotic promoters using hidden Markov models. In: **Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology (ISMB-96)**, p. 182-191, 1996.

Python Software Foundation. Disponível em www.python.org, último acesso em 04 de maio de 2006.

R Development Core Team. **R**: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria, 2005. URL http://www.R-project.org.

REIS, Adriana Neves dos; LEMKE, Ney. Aplicando HMMs no reconhecimento de regiões promotoras em genomas procarióticos. In: **Anais do I WorkComp Sul**, mídia digital, 2004.

REIS, Adriana Neves dos. **Reconhecimento e Predição de Promotores Procarióticos: investigação de uma metodologia *in silico* baseada em HMMs.** 2005. 114 f. Dissertação (Mestrado em Computação Aplicada) – Programa Interdisciplinar de Pós-Graduação em Computação Aplicada (PIPCA), [2005].

RUMELHART, David E; HINTON, Geoffrey E.; WILLIAMS, RONALD J Learning representations by back-propagating errors. **Nature**, v 323, n 6088 p. 533-536, 1986.

RUSSELL, Stuart J.; NORVIG, Peter. **Artificial intelligence:** a modern approach. 2. ed. New Jersey: Prentice Hall International, c2003. 1080 p.

SALGADO, Heladia; GAMA-CASTRO, Socorro; PERALTA-GIL, Martín; DIAZ-PEREDO, Edgar; SÁNCHEZ-SOLANO, Fabiola; SANTOS-ZAVALETA, Alberto; MARTINEZ-FLORES, Irma; JIMENEZ-JACINTO, Verónica; BONAVIDES-MARTINEZ, César; SEGURA-SALAZAR, Juan; MARTINEZ-ANTONIO, Agustino; COLLADO-VIDES, Julio. RegulonDB (version 5.0): Escherichia coli K-12 transcriptional regulatory network, operon organization, and growth conditions. **Nucleic Acids Research**. v. 34, p. 394-397, 2006.

SANTALUCIA, John Jr; HICKS, Donald. The thermodynamics of DNA Structural Motifs. **Annual Review of Biophysics and Biomolecular Structure**, v. 33, p. 415-440, 2004.

SANTALUCIA, John Jr. A unified view of polymer, dumbbell and oligonucleotid DNA nearst-neighbor thermodynamics. **Proceedings of The National Academy of Sciences USA**, v. 95, p.1460-1465, 1998.

SIVARAMAN, Karthikeyan; SESHASAYEE, Aswin Sai Narain; SWAMINATHAN, Krishnakumar; MUTHUKUMARAN, Geetha; PENNATHUR, Gautam. Promoter addresses: revelations from oligonucleotide profiling applied to the *Escherichia coli* genome. **Theoretical Biology an Medical Modelling**, v. 2, 2005.

SPSS. SPSS Inc. Disponível em www.spss.com. Último acesso em 08 de maio de 2006.

THOMPSON, Julie D.; HIGGINS, Desmond D.; GIBSON, Toby J. Clustal W: 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, p. 4673-4680, 1994.

THOMPSON, Julie D.; GIBSON, Toby J.; PLEWNIAK, Frédéric; JEANMOUGIN, François; HIGGINS, Desmond D. The Clustal_X windows interface: flexible strategies for multiple sequence alignment aided by quality analysis tools. **Nucleic Acids Research**, v. 25, n. 24, p. 4876-4882, 1997.

WESTPHAL, Christopher; BLAXTON, Teresa. **Data mining solutions**: methods and tools for solving real-world problems. New York: John Wiley & Sons, 1998. 617 p.

WHEELER, David L.; BARRET, Tanya; BENSON, Dennis A.; BRYANT, Stephen H.; Canese, Kathi; CHETVERNIN, Vyacheslav; CHURCH, Deanna M.; DICUCCIO, Michael; EDGAR, Ron; FEDERHEN, Scott; GEER, Lewis Y.; HELMBERG, Wolfgang; KAPUSTIN, Yuri; KENTON, David L.; KHOVAYKO, Oleg; LIPMAN, David J.; MADDEN, Thomas L.; MAGLOTT, James Ostell; PRUITT, Kim D.; SCHULER, Gregory D.; SCHRIML, Lynn M.; SEQUEIRA, Edwin; SHERRY, Stephen T.; SIROTKIN, Karl; SOUVOROV, Alexandre; STARCHENKO, Grigory; SUZEK, Tubga O.; TATUSOV, Roman; TATUSOVA, Tatiana A.; WAGNER, Lukas; YASCHENKO, Eugene. Database resources of the National Center for Biotechnology Information. **Nucleic Acids Research,** v. 34, p. 173-180, 2006.

WITTEN, Ian H.; FRANK, Eibe. **Data Mining: Practical machine learning tools and techniques.** San Francisco: Morgan Kaufman, 2005. 525 p.

WU, Cathy H. Artificial neural networks for molecular sequence analysis. **Computers & Chemistry,** v. 21, n. 4, p. 237-256, 1997.

WU, Cathy H.; MCLARTY, Jerry Wayne. **Neural networks and genome informatics.** New York: Elsevier, 2000. 205 p.

XING, Ke; DENG, Riqiang; WANG, Jinwen; FENG, Jinghua; HUANG, Mingsong; WANG, Xunzhang. Analysis and prediction of baculovirus promoter sequences. **Virus Research,** v. 113, n. 1, p. 64-71, 2005.

ANEXO A – Matrizes de confusão média obtidas para as arquiteturas da **Simulação 1** com 324 neurônios na camada de entrada e com os dados sem alinhamento prévio.

Tabela A – Matriz de confusão da rede com 324 neurônios de entrada, 1 neurônio na camada oculta e 1 neurônio de saída e 20 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	56	38
Não promotor	16	78

Tabela B – Matriz de confusão da rede com 324 neurônios de entrada, 5 neurônios na camada oculta e 1 neurônio de saída e 20 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	53	41
Não promotor	12	82

Tabela C - Matriz de confusão da rede com 324 neurônios de entrada, 10 neurônios na camada oculta e 1 neurônio de saída e 25 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	49	45
Não promotor	11	83

Tabela D - Matriz de confusão da rede com 324 neurônios de entrada, 15 neurônios na camada oculta e 1 neurônio de saída e 25 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	56	38
Não promotor	10	84

Tabela E - Matriz de confusão da rede com 324 neurônios de entrada, 20 neurônios na camada oculta e 1 neurônio de saída e 25 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	57	37
Não promotor	12	82

ANEXO B – Matrizes de confusão média obtidas para as arquiteturas da **Simulação 1** com 284 neurônios na camada de entrada e com os dados sem alinhamento prévio.

Tabela A - Matriz de confusão da rede com 284 neurônios de entrada, 1 neurônio na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	51	43
Não promotor	14	80

Tabela B - Matriz de confusão da rede com 284 neurônios de entrada, 5 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	49	45
Não promotor	11	83

Tabela C - Matriz de confusão da rede com 284 neurônios de entrada, 10 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	50	44
Não promotor	11	83

Tabela D- Matriz de confusão da rede com 284 neurônios de entrada, 15 neurônios na camada oculta e 1 neurônio de saída e 50 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	58	36
Não promotor	13	81

ANEXO C – Matrizes de confusão média obtidas para as arquiteturas da **Simulação 1** com 244 neurônios na camada de entrada e com os dados sem alinhamento prévio.

Tabela A - Matriz de confusão da rede com 244 neurônios de entrada, 1 neurônio na camada oculta e 1 neurônio de saída e 20 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	52	42
Não promotor	14	80

Tabela B - Matriz de confusão da rede com 244 neurônios de entrada, 3 neurônios na camada oculta e 1 neurônio de saída e 20 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	48	46
Não promotor	12	82

Tabela C - Matriz de confusão da rede com 244 neurônios de entrada, 5 neurônios na camada oculta e 1 neurônio de saída e 20 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	53	41
Não promotor	14	70

Tabela D - Matriz de confusão da rede com 244 neurônios de entrada, 10 neurônios na camada oculta e 1 neurônio de saída e 20 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	53	41
Não promotor	12	82

ANEXO D – Matrizes de confusão média obtidas para as arquiteturas da **Simulação 1** com 288 neurônios na camada de entrada e com os dados previamente alinhados.

Tabela A - Matriz de confusão da rede com 288 neurônios de entrada, 1 neurônio na camada oculta e 1 neurônio de saída e 50 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	65	68
Não promotor	16	78

Tabela B - Matriz de confusão da rede com 288 neurônios de entrada, 3 neurônios na camada oculta e 1 neurônio de saída e 30 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	70	24
Não promotor	14	70

Tabela C - Matriz de confusão da rede com 288 neurônios de entrada, 5 neurônios na camada oculta e 1 neurônio de saída e 30 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	72	22
Não promotor	13	81

ANEXO E – Matrizes de confusão média obtidas para as arquiteturas da **Simulação 2** com 80 neurônios na camada de entrada e com os valores de estabilidade sem suavização.

Tabela A - Matriz de confusão da rede com 80 neurônios de entrada, 1 neurônio na camada oculta e 1 neurônio de saída e 50 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	75	19
Não promotor	55	39

Tabela B - Matriz de confusão da rede com 80 neurônios de entrada, 2 neurônios na camada oculta e 1 neurônio de saída e 50 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	70	24
Não promotor	43	51

Tabela C - Matriz de confusão da rede com 80 neurônios de entrada, 3 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	60	34
Não promotor	35	59

Tabela D - Matriz de confusão da rede com 80 neurônios de entrada, 4 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	62	32
Não promotor	31	63

Tabela E - Matriz de confusão da rede com 80 neurônios de entrada, 5 neurônios na camada oculta e 1 neurônio de saída e 30 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	62	32
Não promotor	28	66

Tabela F - Matriz de confusão da rede com 80 neurônios de entrada, 6 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	62	32
Não promotor	28	66

Tabela G - Matriz de confusão da rede com 80 neurônios de entrada, 7 neurônios na camada oculta e 1 neurônio de saída e 30 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	61	33
Não promotor	30	64

Tabela H - Matriz de confusão da rede com 80 neurônios de entrada, 8 neurônios na camada oculta e 1 neurônio de saída e 30 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	60	34
Não promotor	30	64

ANEXO F – Matrizes de confusão média obtidas para as arquiteturas da **Simulação 2** com 79 neurônios na camada de entrada e com os valores de estabilidade sem suavização.

Tabela A - Matriz de confusão da rede com 79 neurônios de entrada, 1 neurônio na camada oculta e 1 neurônio de saída e 50 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	83	11
Não promotor	71	20

Tabela B - Matriz de confusão da rede com 79 neurônios de entrada, 2 neurônios na camada oculta e 1 neurônio de saída e 60 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	61	33
Não promotor	38	56

Tabela C - Matriz de confusão da rede com 79 neurônios de entrada, 3 neurônios na camada oculta e 1 neurônio de saída e 50 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	64	30
Não promotor	42	52

Tabela D - Matriz de confusão da rede com 79 neurônios de entrada, 4 neurônios na camada oculta e 1 neurônio de saída e 60 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	60	34
Não promotor	32	62

Tabela E - Matriz de confusão da rede com 79 neurônios de entrada, 5 neurônios na camada oculta e 1 neurônio de saída e 60 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	60	34
Não promotor	30	64

Tabela F- Matriz de confusão da rede com 79 neurônios de entrada, 6 neurônios na camada oculta e 1 neurônio de saída e 30 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	57	37
Não promotor	33	61

Tabela G - Matriz de confusão da rede com 79 neurônios de entrada, 7 neurônios na camada oculta e 1 neurônio de saída e 20 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	52	42
Não promotor	29	65

Tabela H - Matriz de confusão da rede com 79 neurônios de entrada, 8 neurônios na camada oculta e 1 neurônio de saída e 20 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	52	42
Não promotor	30	64

ANEXO G – Matrizes de confusão média obtidas para as arquiteturas da **Simulação 2** com 78 neurônios na camada de entrada e com os valores de estabilidade sem suavização.

Tabela A - Matriz de confusão da rede com 78 neurônios de entrada, 1 neurônio na camada oculta e 1 neurônio de saída e 60 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	82	12
Não promotor	69	25

Tabela B - Matriz de confusão da rede com 78 neurônios de entrada, 2 neurônios na camada oculta e 1 neurônio de saída e 80 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	78	16
Não promotor	67	27

Tabela C - Matriz de confusão da rede com 78 neurônios de entrada, 3 neurônios na camada oculta e 1 neurônio de saída e 80 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	68	26
Não promotor	49	45

Tabela D - Matriz de confusão da rede com 78 neurônios de entrada, 4 neurônios na camada oculta e 1 neurônio de saída e 30 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	58	36
Não promotor	40	54

Tabela E - Matriz de confusão da rede com 78 neurônios de entrada, 5 neurônios na camada oculta e 1 neurônio de saída e 20 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	61	33
Não promotor	45	49

Tabela F - Matriz de confusão da rede com 78 neurônios de entrada, 6 neurônios na camada oculta e 1 neurônio de saída e 20 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	57	37
Não promotor	35	59

Tabela G - Matriz de confusão da rede com 78 neurônios de entrada, 7 neurônios na camada oculta e 1 neurônio de saída e 50 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	63	31
Não promotor	38	56

Tabela H - Matriz de confusão da rede com 78 neurônios de entrada, 8 neurônios na camada oculta e 1 neurônio de saída e 30 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	56	38
Não promotor	34	60

ANEXO H – Matrizes de confusão média obtidas para as arquiteturas da **Simulação 2** com 76 neurônios na camada de entrada e com os valores de estabilidade sem suavização.

Tabela A - Matriz de confusão da rede com 76 neurônios de entrada, 1 neurônio na camada oculta e 1 neurônio de saída e 20 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	85	9
Não promotor	74	20

Tabela B - Matriz de confusão da rede com 76 neurônios de entrada, 2 neurônios na camada oculta e 1 neurônio de saída e 30 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	80	14
Não promotor	75	19

Tabela C - Matriz de confusão da rede com 76 neurônios de entrada, 3 neurônios na camada oculta e 1 neurônio de saída e 60 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	65	68
Não promotor	46	48

Tabela D - Matriz de confusão da rede com 76 neurônios de entrada, 4 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	70	24
Não promotor	53	41

Tabela E - Matriz de confusão da rede com 76 neurônios de entrada, 5 neurônios na camada oculta e 1 neurônio de saída e 20 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	64	30
Não promotor	42	52

Tabela F - Matriz de confusão da rede com 76 neurônios de entrada, 6 neurônios na camada oculta e 1 neurônio de saída e 30 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	63	31
Não promotor	40	54

Tabela G - Matriz de confusão da rede com 76 neurônios de entrada, 7 neurônios na camada oculta e 1 neurônio de saída e 30 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	57	37
Não promotor	39	55

Tabela H - Matriz de confusão da rede com 76 neurônios de entrada, 8 neurônios na camada oculta e 1 neurônio de saída e 30 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	58	36
Não promotor	38	56

ANEXO I – Matrizes de confusão média obtidas para as arquiteturas da **Simulação 2** com 80 neurônios na camada de entrada e com os valores de estabilidade suavizados com grau 3.

Tabela A - Matriz de confusão da rede com 80 neurônios de entrada, 1 neurônio na camada oculta e 1 neurônio de saída e 60 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	78	16
Não promotor	55	39

Tabela B - Matriz de confusão da rede com 80 neurônios de entrada, 2 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	72	22
Não promotor	40	54

Tabela C - Matriz de confusão da rede com 80 neurônios de entrada, 3 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	70	24
Não promotor	38	56

Tabela D - Matriz de confusão da rede com 80 neurônios de entrada, 5 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	65	29
Não promotor	27	67

Tabela E - Matriz de confusão da rede com 80 neurônios de entrada, 6 neurônios na camada oculta e 1 neurônio de saída e 30 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	65	29
Não promotor	26	68

Tabela F - Matriz de confusão da rede com 80 neurônios de entrada, 7 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	65	29
Não promotor	25	69

Tabela G - Matriz de confusão da rede com 80 neurônios de entrada, 8 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	65	29
Não promotor	26	68

ANEXO J – Matrizes de confusão média obtidas para as arquiteturas da **Simulação 2** com 80 neurônios na camada de entrada e com os valores de estabilidade suavizados com grau 5.

Tabela A - Matriz de confusão da rede com 80 neurônios de entrada, 1 neurônio na camada oculta e 1 neurônio de saída e 60 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	83	11
Não promotor	60	34

Tabela B - Matriz de confusão da rede com 80 neurônios de entrada, 2 neurônios na camada oculta e 1 neurônio de saída e 50 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	69	25
Não promotor	32	62

Tabela C - Matriz de confusão da rede com 80 neurônios de entrada, 3 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	60	34
Não promotor	22	72

Tabela D - Matriz de confusão da rede com 80 neurônios de entrada, 5 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	65	39
Não promotor	25	69

Tabela E - Matriz de confusão da rede com 80 neurônios de entrada, 6 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	66	28
Não promotor	24	70

Tabela F - Matriz de confusão da rede com 80 neurônios de entrada, 7 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	65	29
Não promotor	24	70

Tabela G - Matriz de confusão da rede com 80 neurônios de entrada, 8 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	66	28
Não promotor	26	68

ANEXO L – Matrizes de confusão média obtidas para as arquiteturas da **Simulação 2** com 80 neurônios na camada de entrada e com os valores de estabilidade suavizados com grau 7.

Tabela A - Matriz de confusão da rede com 80 neurônios de entrada, 1 neurônio na camada oculta e 1 neurônio de saída e 70 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	77	17
Não promotor	48	46

Tabela B - Matriz de confusão da rede com 80 neurônios de entrada, 2 neurônios na camada oculta e 1 neurônio de saída e 60 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	79	15
Não promotor	52	42

Tabela C - Matriz de confusão da rede com 80 neurônios de entrada, 3 neurônios na camada oculta e 1 neurônio de saída e 50 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	65	29
Não promotor	25	69

Tabela D - Matriz de confusão da rede com 80 neurônios de entrada, 4 neurônios na camada oculta e 1 neurônio de saída e 50 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	63	31
Não promotor	24	70

Tabela E - Matriz de confusão da rede com 80 neurônios de entrada, 5 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	66	28
Não promotor	26	68

Tabela F - Matriz de confusão da rede com 80 neurônios de entrada, 6 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	67	27
Não promotor	24	70

Tabela G - Matriz de confusão da rede com 80 neurônios de entrada, 7 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	65	29
Não promotor	20	74

Tabela H - Matriz de confusão da rede com 80 neurônios de entrada, 8 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	66	28
Não promotor	22	72

ANEXO M – Matrizes de confusão média obtidas para as arquiteturas da **Simulação 2** com 80 neurônios na camada de entrada e com os valores de estabilidade suavizados com grau 9.

Tabela A - Matriz de confusão da rede com 80 neurônios de entrada, 1 neurônio na camada oculta e 1 neurônio de saída e 80 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	81	13
Não promotor	60	34

Tabela B - Matriz de confusão da rede com 80 neurônios de entrada, 2 neurônios na camada oculta e 1 neurônio de saída e 50 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	70	24
Não promotor	34	60

Tabela C - Matriz de confusão da rede com 80 neurônios de entrada, 3 neurônios na camada oculta e 1 neurônio de saída e 50 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	68	26
Não promotor	28	66

Tabela D - Matriz de confusão da rede com 80 neurônios de entrada, 5 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	67	27
Não promotor	21	73

Tabela E - Matriz de confusão da rede com 80 neurônios de entrada, 6 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	68	26
Não promotor	28	66

Tabela F - Matriz de confusão da rede com 80 neurônios de entrada, 7 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	67	27
Não promotor	23	71

Tabela G - Matriz de confusão da rede com 80 neurônios de entrada, 8 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	66	28
Não promotor	22	72

ANEXO N – Matrizes de confusão média obtidas para as arquiteturas da **Simulação 2** com 80 neurônios na camada de entrada e com os valores de estabilidade suavizados com grau 15.

Tabela A - Matriz de confusão da rede com 80 neurônios de entrada, 1 neurônio na camada oculta e 1 neurônio de saída e 100 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	75	19
Não promotor	42	52

Tabela B - Matriz de confusão da rede com 80 neurônios de entrada, 2 neurônios na camada oculta e 1 neurônio de saída e 70 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	73	21
Não promotor	42	52

Tabela C - Matriz de confusão da rede com 80 neurônios de entrada, 3 neurônios na camada oculta e 1 neurônio de saída e 60 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	70	24
Não promotor	30	64

Tabela D - Matriz de confusão da rede com 80 neurônios de entrada, 4 neurônios na camada oculta e 1 neurônio de saída e 60 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	67	27
Não promotor	25	69

Tabela E - Matriz de confusão da rede com 80 neurônios de entrada, 5 neurônios na camada oculta e 1 neurônio de saída e 50 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	66	28
Não promotor	24	70

Tabela F - Matriz de confusão da rede com 80 neurônios de entrada, 6 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	66	28
Não promotor	22	72

Tabela G - Matriz de confusão da rede com 80 neurônios de entrada, 7 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	67	27
Não promotor	22	72

Tabela H - Matriz de confusão da rede com 80 neurônios de entrada, 8 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	66	28
Não promotor	22	72

ANEXO O – Matrizes de confusão média obtidas para as arquiteturas da **Simulação 2** com 80 neurônios na camada de entrada e com os valores de estabilidade suavizados com grau 20.

Tabela A - Matriz de confusão da rede com 80 neurônios de entrada, 1 neurônio na camada oculta e 1 neurônio de saída e 90 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	72	22
Não promotor	33	61

Tabela B - Matriz de confusão da rede com 80 neurônios de entrada, 2 neurônios na camada oculta e 1 neurônio de saída e 70 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	79	15
Não promotor	47	47

Tabela C - Matriz de confusão da rede com 80 neurônios de entrada, 3 neurônios na camada oculta e 1 neurônio de saída e 60 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	71	23
Não promotor	30	64

Tabela D - Matriz de confusão da rede com 80 neurônios de entrada, 4 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	67	27
Não promotor	24	70

Tabela E - Matriz de confusão da rede com 80 neurônios de entrada, 5 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	67	27
Não promotor	20	74

Tabela F - Matriz de confusão da rede com 80 neurônios de entrada, 6 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	66	28
Não promotor	24	70

Tabela G - Matriz de confusão da rede com 80 neurônios de entrada, 7 neurônios na camada oculta e 1 neurônio de saída e 50 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	68	26
Não promotor	23	71

Tabela H - Matriz de confusão da rede com 80 neurônios de entrada, 8 neurônios na camada oculta e 1 neurônio de saída e 40 épocas de aprendizado.

	Classificado como promotor	Classificado como não-promotor
Promotor	68	26
Não promotor	22	72