JÔNATAS LOPES DE PAIVA

ESTUDO DE APLICAÇÃO DE UM MÉTODO HÍBRIDO PARA CLASSIFICAÇÃO DE PROTEÍNAS UTILIZANDO SEQUENCE CODING BY SLIDING WINDOW E REDES NEURAIS ARTIFICIAIS

LAVRAS - MG
2010

JÔNATAS LOPES DE PAIVA

ESTUDO DE APLICAÇÃO DE UM MÉTODO HÍBRIDO PARA CLASSIFICAÇÃO DE PROTEÍNAS UTILIZANDO SEQUENCE CODING BY SLIDING WINDOW E REDES NEURAIS ARTIFICIAIS

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

Orientador:

M.Sc. Cristiano Leite de Castro

LAVRAS - MG
2010

JÔNATAS LOPES DE PAIVA

ESTUDO DE APLICAÇÃO DE UM MÉTODO HÍBRIDO PARA CLASSIFICAÇÃO DE PROTEÍNAS UTILIZANDO SEQUENCE CODING BY SLIDING WINDOW E REDES NEURAIS ARTIFICIAIS

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

APROVADA em ____ de _____ de _____

Dr. Ahmed Ali Abdalla Esmin UFLA

Dra. Patrícia Gomes Cardoso UFLA

M.Sc. Cristiano Leite de Castro

Orientador

LAVRAS - MG
2010

Aos meus tios Edvaldo e Edmilsom.

A meu pai José Heitor.

A minha avó Maura.

A meu avô David e tia Maria (in memoriam).

A minha irmã Lara.

A minha mãe, Alcione.

DEDICO.

AGRADECIMENTOS

Agradeço ao professor e orientador Thiago, que confiou em mim e deu a oportunidade de começar este projeto.

Agradeço ao professor Cristiano, que aceitou me orientar durante a parte crítica deste trabalho, e ajudou sempre que eu precisei.

Ao professor André Saúde, que deu a chance de poder terminar este trabalho mesmo enquanto orientado dele.

A todos os outros professores do DCC – UFLA, que durante estes quase quatro anos conseguiram fazer com que eu aprendesse muito.

A meus amigos de curso, que estiveram presentes nas horas de alegria e nas horas difíceis. Pra um bando de "nerdjonhsons" vocês são todos sensacionais.

Aos meus companheiros de trabalho na Mitah, que ao longo do último ano se mostraram grandes amigos e companheiros, até nos finais de sprint.

A minha família. Agradeço a todos vocês, os que estão próximos e os que estão distantes, os que estão conosco e os que já se foram, vocês são os responsáveis por me fazer chegar aqui.

Em especial a minha mãe Alcione. Que me deu apoio em todas as decisões que eu tomei. Que me aconselhou sempre que eu tive dúvidas. Que me deu suporte sempre que eu tive dificuldades.

RESUMO

Este trabalho propõe a construção de um classificador para as proteínas do banco de dados público COG (*Clusters of Orthologous Groups*). O classificador é construído utilizando redes neurais artificiais (RNA). O grande problema deste método é que RNAs aceitam entradas de tamanho único, e as proteínas do COG possuem tamanhos variados, por este motivo as proteínas precisam ser codificadas, e a codificação utilizada neste trabalho é a *Sequence Coding By Sliding Window* (SCSW), que se utiliza de janelas deslizantes e gera vetores de tamanho único para qualquer entrada. O empecilho desta codificação é o fato de que os vetores gerados podem ser ambíguos, fazendo com que sequências diferentes gerem vetores iguais, por este motivo um tamanho ideal para a janela deslizante deve ser encontrado. Com a codificação pronta, os conjuntos de dados para treinamento e validação da RNA devem ser selecionados, essa seleção deve ser feita porque os dados obtidos do COG possuem dimensionalidade muito grande, o que torna uma seleção de dados necessária. A seleção é feita utilizando o método *Fuzzy c-means*, que seleciona os pontos e cria os conjuntos para a criação da RNA. Ao final, os resultados obtidos com a RNA no conjunto de validação são comparados aos resultados esperados, com esses resultados o índice de acerto da rede é calculado.

Palavras-chave**:** Classificação de proteínas. Codificação de proteínas. Redes neurais artificiais. *Clustering*; Bioinformática.

ABSTRACT

This work proposes the construction of a protein classifier for the public database COG (Clusters of Orthologous groups). The classifier is built using artificial neural networks (ANN). The major problem with this method is that ANNs accept only single sizes inputs, and the proteins in COG can have many different sizes, because of that the proteins need to be coded, and the coding method used in this work is the Sequence Coding By Sliding Window (SCSW), which uses sliding windows and generates unique size vectors for any entry. The downside with this coding is that the vectors generated can be ambiguous, causing different sequences to generate equal vectors, to avoid that an ideal size for the sliding window must be found. With the encoding ready the datasets for training and validation of ANN must be selected, this selection must be done because the data obtained from COG are too large; and this makes a selection of data required. The selection is done using the Fuzzy c-means method; it selects the points and creates sets for the creation of ANN. In the end, the results obtained with the ANN in the validation set are compared with the expected results, with these results the hit rate of the network is calculated.

Keywords***:*** Protein classification; Protein coding; Artificial neural networks; Clustering; Bioinformatics.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS

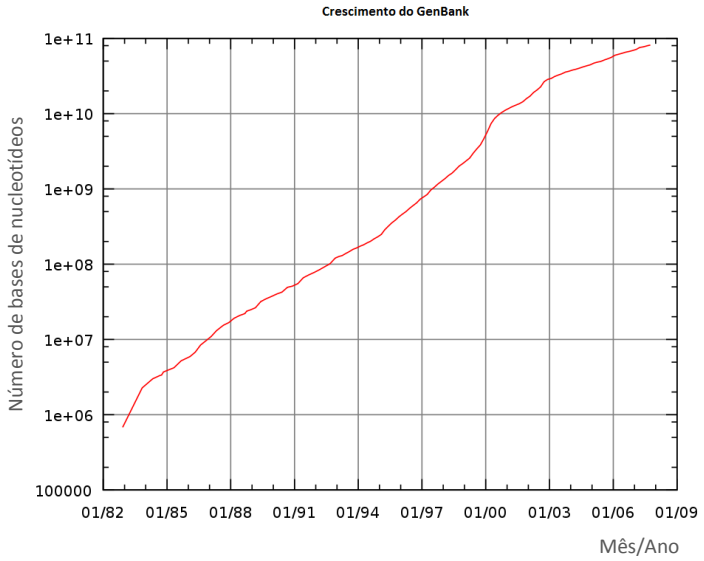
SCSW	*Sequence Coding By Sliding Window*
FCM	*Fuzzy c-means*
RNA	Rede Neural Artificial
COG	*Clusters of Orthologous Groups*

SUMÁRIO

1 INTRODUÇÃO

1.1 Contextualização e Motivação

Diversos bancos de dados públicos armazenam sequências de DNA como GenBank[1], EMBL (*European Molecular Biology Laboratory*)[2], DDBJ (DNA *Data Bank of Japan*)[3], PIR (*Protein Information Research*)[4], Swiss-Prot (*Protein knowledgebase*)[5], Smart (*Simple Modular Architecture Research Tool*)[6], CDD (*Conserved Domain Database*)[7], COG (*Clusters of Orthologous Groups*)[8], entre outros. Além dos bancos de dados, métodos computacionais para recuperação e análise de dados foram e ainda estão em continuo desenvolvimento.



[1] http://www.ncbi.nlm.nih.gov/Genbank/
[2] http://www.ebi.ac.uk/embl/
[3] http://www.ddbj.nig.ac.jp/
[4] http://pir.georgetown.edu/
[5] http://ca.expasy.org/sprot/
[6] http://smart.embl-heidelberg.de/
[7] http://www.ncbi.nlm.nih.gov/Structure/cdd/cdd.shtml
[8] http://www.ncbi.nlm.nih.gov/COG/

Figura 1 Número de bases de nucleotídeos no GenBank em relação ao mês e ano.

A Figura 1 mostra como o banco de dados GenBank cresceu entre 1982 e 2009, nela podemos ver que a quantidade de bases de nucleotídeos e seqüências vem aumentando de maneira quase exponencial nos últimos anos, e métodos computacionais para classificação desses dados se mostra cada vez mais importantes.

A classificação das proteínas é feita se baseando na ocorrência de padrões de aminoácidos definidos em seus domínios. Essa classificação deve ser confirmada em laboratório, o que demanda tempo e recurso. Logo, os métodos computacionais devem ser precisos em seus resultados, de modo que a confirmação laboratorial seja direcionada pelos resultados desses métodos.

As proteínas são descritas sobre até quatro aspectos no que diz respeito à estrutura (estrutura primária, secundária, terciária e, em alguns casos, quaternária), mas os métodos computacionais normalmente realizam essa comparação a partir de suas estruturas primárias (neste trabalho, sempre que a palavra "proteína" for mencionada estaremos nos referindo à seqüência de aminoácidos que formam a proteína).

Normalmente os métodos computacionais utilizados para classificar proteínas por comparação são baseados nos métodos de alinhamento par-a-par. Mas muitas seqüências nos bancos de dados públicos ainda não foram classificadas ou foram classificadas de maneira inconsistente.

Redes neurais artificiais (RNA) são modelos computacionais capazes de generalizar informação após passar por um treinamento específico, e a utilização de RNAs para a construção de classificadores é uma abordagem amplamente aceita. Desta maneira, um sistema de classificação de proteínas utilizando RNAs pode ser criado de modo a ser um complemento aos métodos baseados no alinhamento par-a-par.

Este trabalho propõe a utilização de RNAs para a classificação de proteínas de acordo com sua classe funcional, para isso ele se utiliza do banco de dados público COG, que possuí diversos conjuntos de proteínas agrupadas de acordo com suas funções[9].

1.2 Objetivos do Trabalho

1.2.1 Objetivo Geral

O objetivo deste trabalho é a implementação de um classificador de proteínas, utilizando redes neurais artificiais, baseado nas classes funcionais do COG.

1.2.2 Objetivos Específicos

O trabalho ainda possuí objetivos específicos, que são:

- Implementar a metodologia descrita em (Pevzner 1995), para definir o tamanho ideal de janela para a utilização do SCSW em um dado conjunto de seqüências.
- Coletar a base de dados de proteínas do COG para a aplicação da metodologia descrita no tópico anterior.
- Com a base de dados coletada, aplicar um método baseado no algoritmo *Fuzzy K-Means* para criar um conjunto de dados para treinamento e um para validação da RNA.

[9] http://www.ncbi.nlm.nih.gov/COG/old/palox.cgi?fun=all

1.3 Estrutura do Trabalho

Este trabalho está estruturado da seguinte maneira:

- O Capítulo 2 apresenta o Referencia Teórico, mostrando os principais conceitos utilizados no trabalho.
- O Capítulo 3 mostra a Metodologia do trabalho, detalhando a seqüência de atividades, a maneira de condução do trabalho e o modo como os conceitos foram aplicados.
- O Capítulo 4 tem os Resultados deste trabalho, mostrando quais foram os resultados finais.
- O Capítulo 5 finaliza o trabalho apresentando as conclusões, discutindo os resultados obtidos e propondo trabalhos futuros.

2 REFERENCIAL TEÓRICO

2.1 Esquema de codificação de seqüências com *Sequence Coding By Sliding Window*

Apesar de uma proteína poder ser descrita sobre até quatro aspectos relacionados à sua estrutura, apenas a estrutura primária é considerada para as operações de comparação. A estrutura primária é dada pela seqüência de aminoácidos que compõem a proteína, e determina todo o arranjo e especificidade da proteína.

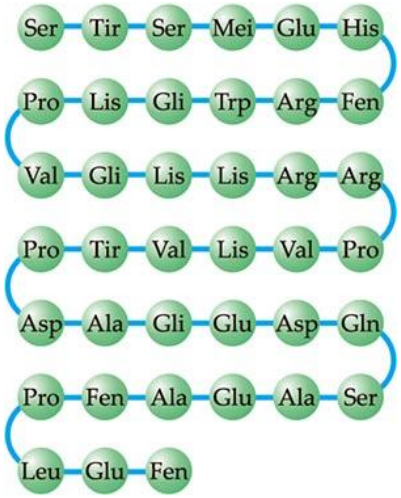


Figura 2 Seqüência de aminoácidos, exemplificando a estrutura primária de uma proteína hipotética.

A Figura 2 representa uma seqüência de aminoácidos, neste caso representando a estrutura primária de uma proteína hipotética.

As estruturas primárias das proteínas podem ser representadas como *strings* de caracteres, onde cada caractere representa um aminoácido, de acordo com sua abreviatura de uma letra.

2.1.1 Comparação de seqüências

Na análise de proteínas por métodos computacionais, a comparação de seqüências é considerada uma operação básica.

Alguns dos principais métodos computacionais aplicados em bioinformática são os algoritmos de alinhamento de seqüências, que procuram alinhar duas cadeias de caracteres e analisar se elas possuem similaridades. O principal método de alinhamento de seqüências é chamado de alinhamento par-a-par (Altschul *et al.*, 1990).

K	G	S	S	**R**	I	W	**D**	N
K	S	A	G	**R**	L	G	**D**	A

Figura 3 Alinhamento entre duas seqüências, onde os acertos estão em negrito

A Figura 3 exemplifica o funcionamento de um algoritmo de alinhamento de seqüências. Duas seqüências são comparadas caractere por caractere, marcando o número de acertos encontrados (neste caso os acertos estão em negrito). Desta maneira, estes algoritmos tentam encontrar o melhor alinhamento possível entre duas seqüências.

2.1.2 Uso do SCSW

Uma maneira alternativa aos algoritmos de alinhamento para a classificação de proteínas são as Redes Neurais Artificiais. O maior obstáculo a utilização das RNAs é: depois de treinadas elas só aceitam entradas de tamanho único e cada *string* de caracteres que representa uma proteína possuí um tamanho diferente.

Como meio de resolver este problema, as seqüências devem ser codificadas, de maneira que todas as seqüências possuam tamanhos iguais.

O método *Sequence Coding by Sliding Window* (SCSW) (Blaisdell, 1986) consiste em transformar cada cadeia de caracteres em um vetor, onde cada posição do vetor representa uma combinação de elementos do alfabeto da cadeia de caracteres. De acordo com Rodrigues (2007), essa codificação pode ser feita da seguinte maneira:

- Seja a seqüência S uma seqüência qualquer de tamanho N, definida sobre um alfabeto de tamanho a.
- Uma janela deslizante w_n, de tamanho $1 \leq n \leq N$ é colocada no inicio de S e "desliza", uma posição por vez, até a posição $N - n + 1$ da seqüência.
- Um vetor V_n, de tamanho a^n, é definido, onde cada posição corresponde a uma possível *n-tupla* dos elementos de a.
- A cada passo do SCSW, o elemento de V_n que corresponde à subseqüência em que a janela se encontra é incrementado.
- Quando w_n chegar atingir a posição final, o vetor V_n conterá a quantidade de vezes que cada combinação de elementos de a foi encontrada em S. O vetor V_n sempre possuirá o mesmo tamanho para um mesmo tamanho de janela, independente do tamanho da seqüência.

Quadro 1 SCSW, com janela deslizante w_n de tamanho $n = 3$ e alfabeto $a = \{ A, B \}$, aplicado à seqüência S=ABAAB

AAA	AAB	ABA	ABB	BAA	BAB	BBA	BBB
0	**1**	**1**	0	**1**	0	0	0

O Quadro 1 mostra como seria a codificação de uma seqüência S = "ABAAB" utilizando o SCSW para uma janela w_n de tamanho $n = 3$ e alfabeto $a = \{ A, B \}$. No início do método, a janela w corresponde à subseqüência "ABA",

dessa maneira a posição no vetor que corresponde a *w* tem seu valor incrementado; o método continua até que cada subseqüência de *S* seja visitada. O resultado é um vetor final de tamanho $|a|^n$ (neste exemplo é 3^2), onde cada posição do vetor tem o número de vezes que determinada cadeia de caracteres foi encontrada em *S*.

O SCSW pode apresentar problemas quando aplicado, gerando vetores iguais para seqüências diferentes. Deste modo, o tamanho ideal de janela deve ser determinado, de forma que as ambigüidades sejam minimizadas.

A influência do tamanho da janela na codificação pode ser vista no exemplo a seguir. A Figura 4 mostra quatro seqüências diferentes definidas sobre o alfabeto a = {A, B, C}, quando utilizada uma janela de tamanho n = 2 todas as quatro geram o mesmo vetor, como mostrado no Quadro 2.

S_1: A B A A A C A
S_2: A A B A A C A
S_3: A A A B A C A
S_4: A A B A C A A

Figura 4 Diferentes seqüências que geram ambigüidade, para o alfabeto a = {A, B, C}

Quadro 2 SCSW com janela de tamanho 2 aplicado às seqüências descritas na Figura 2.2.

AA	AB	BA	AC	CA
2	1	1	1	1

Por outro lado, o problema da ambigüidade pode ser resolvido quando a janela passa a ter o tamanho n = 3, como visto no Quadro 3.

Quadro 3 SCSW com janela de tamanho 3 aplicado às seqüências da Figura 2.2.

	ABA	BAA	AAA	AAC	ACA	AAB	BAC	CAA
S_1	1	1	1	1	1	0	0	0
S_2	1	1	0	1	1	1	0	0
S_3	1	0	1	0	1	1	1	0
S_4	1	0	0	0	1	1	1	1

O maior desafio da codificação SCSW é encontrar a janela ideal, de maneira que a ambigüidade seja reduzida e os tamanhos dos vetores não aumentem muito. Janelas muito grandes podem se tornar um problema quando utilizadas, por exemplo, um alfabeto de tamanho $|a| = 6$, com uma janela de tamanho $n = 10$ iria gerar vetores de tamanho 6^{10}, o que torna a sua utilização inviável (considerando um vetor de números inteiros, cada um ocupando 16 bits de memória cada, um único vetor de tamanho 6^{10} ocuparia mais de 115 MB de espaço em memória).

2.1.3 Encontrar o tamanho de janela ideal para o SCSW

Em Pevzner (1995) foi mostrado que a ambigüidade entre seqüências tem probabilidade de ocorrer se pelo menos um dos três casos a seguir ocorrer para uma dada janela deslizante *n*:

- Caso 1: S1 = $Y_1\mathbf{Z_1}Y_2\mathbf{Z_2}Y_3\mathbf{Z_1}Y_4\mathbf{Z_2}Y_5$, onde
  Z_1 e Z_2 são (*n*-1)-tuplas que se repetem
  Y_1, Y_3 e Y_5 podem ser vazias;
  Y_2 e Y_4 devem ser diferentes.
  Ex.: A**CGA**AT**CTA**T**CGA**GA**CTA**A
- Caso 2: S2 = $Y_1\mathbf{Z}Y_2\mathbf{Z}Y_3\mathbf{Z}Y_4$, onde
  Z é uma (*n*-1)-tupla que se repete

Y_1 e Y_4 podem ser vazias;

Y_2 e Y_3 devem ser diferentes.

Ex.: AG**CGA**ATC**CGA**AT**CGA**GAA

- Caso 3: S3 = $\mathbf{Z_1}Y_1\mathbf{Z_2}Y_2\mathbf{Z_1}$, onde

  Z_1 e Z_2 são (n-1)-tuplas,

  Y_1 e Y_2 podem ser vazias.

  Ex.: **ATG**CTA**ATG**

Para um dado tamanho de janela, se nenhum dos três casos ocorrer, significa dizer que esta seqüência não gera um vetor ambíguo. Mas, mesmo que alguma cadeia de caracteres apresente algum dos casos, não significa dizer que a ambigüidade irá acontecer.

2.2 *Fuzzy k-means*

O método *Fuzzy c-means* (ou *Fuzzy k-means*, FCM), é um método de reconhecimento de padrões derivado do método *K-means*.

Enquanto o método *K-Means* classifica um elemento como pertencente ou não a algum *cluster*, o FCM indica a probabilidade de um elemento pertencer a dois ou mais *clusters*.

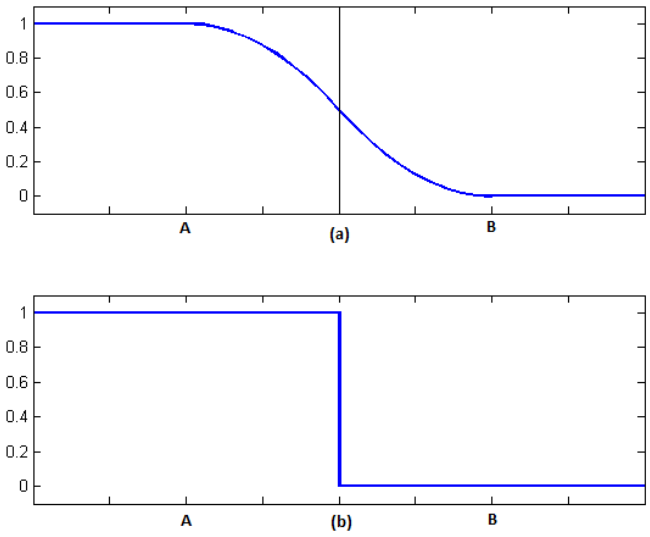


Figura 5 Os gráficos mostram como se portam os algoritmos (a) *Fuzzy c-means* e o (b) *K-means*.

A Figura 5 aponta a diferença entre o FCM e o *K-means*. Em (a) é mostrado o comportamento do FCM, nele cada elemento dentro do domínio tem uma probabilidade de pertencer ao *cluster* A e ao *cluster* B. Em (b) a probabilidade é de 0 ou 100%, ou um elemento pertence ao *cluster* A ou ao *cluster* B, sem probabilidade de pertencer ao outro conjunto.

O FCM é aplicado se baseando na minimização da função (1), que é a função objetivo e está descrita abaixo:

$$J_m = \sum_{i=1}^{N}\sum_{j=1}^{C} u_{ij}^{m} \left\| x_i - c_j \right\|^2, \qquad 1 \leq m \leq \infty \quad (1)$$

Na função (1) m é qualquer número real maior que 1, u_{ij} é a probabilidade de x_i pertencer ao *cluster* j, x_i é o i-ésimo elemento dos dados a ser classificados e c_j é o centróide do *cluster* j.

Os conjuntos são formados com a otimização da função (1) enquanto durante diversas iterações do método. O cálculo da probabilidade de algum elemento pertencer a algum *cluster* é feito segundo a equação (2) e os centróides são calculados utilizando a equação (3).

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\left\| x_i - c_j \right\|}{\left\| x_i - c_k \right\|} \right)^{\frac{2}{m-1}}} \quad (2)$$

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}^{m} \, x_i}{\sum_{i=1}^{N} u_{ij}^{m}} \quad (3)$$

As iterações param quando um critério de parada é atingido, este critério é definido por $\left\{ \left| j^{(k+1)} - j^{(k)} \right| \right\} < \epsilon$, onde ϵ é um critério de parada e $0 < \epsilon \leq 1$, e onde *k* é o número de iterações do algoritmo, conforme mostrado na Figura 6.

- Dada a variável *k*
- Inicialize a matriz *u*
- Faça
    - Incremente *k*
    - Calcule os centróides, de acordo com a equação (3)
    - Atualize a matriz *u*, de acordo com a equação (2)
    - Calcule a função objetivo, de acordo com a equação (1), e armazene em *j*
- Enquanto ($|| j^{(k)} - j^{(k-1)} || < \epsilon$)

Figura 6 Pseudocódigo para o algoritmo *Fuzzy c-means*

Ao final do algoritmo, como mostrado na Figura 6, nós temos na matriz *u* a probabilidade de cada elemento pertencer a cada *cluster*, e, com estes dados, pode-se inferir sobre qual *cluster* cada elemento pertence e a qual distância ele se encontra do centro do *cluster*.

2.3 Redes Neurais Artificiais

De acordo com Braga *et al.*, em sua publicação "Redes Neurais Artificiais: teoria e aplicações", redes neurais artificiais (RNA) podem ser definidas como sendo *"sistemas paralelos distribuídos compostos por unidades de processamento simples (...) dispostas em uma ou mais camadas e interligadas por um grande número de conexões"*. Ele também diz que na maioria dos modelos estas conexões estão associadas a pesos que guardam o conhecimento representado pelo modelo, e que ponderam a entrada recebida por cada neurônio da rede. Essas redes foram criadas para ter funcionamento baseado no cérebro humano.

2.3.1 Neurônio artificial

Warren McCulloch e Walter Pitts foram os primeiros a descrever um modelo para o neurônio artificial. Neste modelo (MCP) o neurônio humano é simplificado para que de acordo com os valores de sua entrada ele retorne uma saída binária, 0 ou 1 (Braga *et al.*, 2000).

No modelo MCP, o neurônio pode ou não disparar (saída igual a 0 ou 1) de acordo com uma "função de ativação". Caso o valor desta função seja maior que um limiar (*treshold*) ele irá disparar.

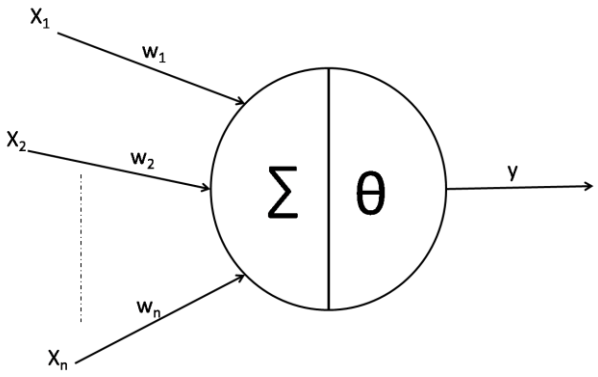


Figura 7 Neurônio de McCulloch e Pitts

A Figura 7 mostra o funcionamento do neurônio no modelo MCP, dado um conjunto de entradas X_i e um conjunto de pesos para essas entradas w_i, a função *y* pode ser calculada com a soma das entradas multiplicadas pelos pesos, caso a saída seja maior que o *treshold* θ, a saída será 1, caso contrário será 0. Além dessa função (Limiar), diversas outras funções de ativação podem ser utilizadas, de maneira que a saída da rede não seja apenas binária e sim um valor difuso.

2.3.2 Aprendizagem da RNA

Para que as RNAs possam classificar os dados de maneira correta ela deve ser treinada por meio de exemplos. Dessa forma o algoritmo de treinamento utilizado deve adaptar os parâmetros da rede para que após um número de iterações do algoritmo haja convergência para uma solução.

O aprendizado pode ser supervisionado, quando a resposta desejada para determinada entrada é informada para a RNA, ou não supervisionado, quando a RNA não possuí uma resposta específica para cada entrada.

Os exemplos mais comuns de algoritmos de treinamento de RNAs supervisionados são a regra delta e o *backpropagation*[10] (que é uma generalização da regra delta para redes de múltiplas camadas). A aprendizagem supervisionada é exemplificada na Figura 8.

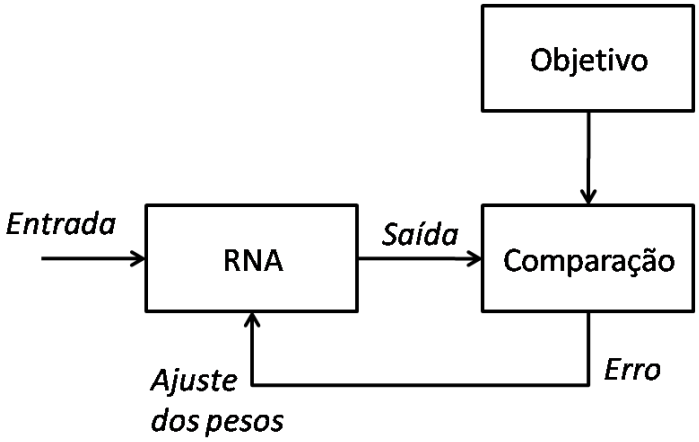


Figura 8 Treinamento supervisionado de uma RNA

2.3.3 Redes *Multilayer Perceptron*

Frank Rosenblatt desenvolveu o modelo *Perceptron* para RNAs em 1957. Este modelo ficou conhecido como *perceptron* de camada única, pois, apesar de possuir três níveis, um de entrada, um intermediário e um de saída, apenas o nível de saída possuía propriedades adaptativas. Em 1962, foi provado que este modelo conseguia atingir a convergência sempre que o problema em questão fosse linearmente separável.

O maior problema do *perceptron* é o fato de não conseguir tratar bem os problemas que não sejam linearmente separáveis.

[10] http://page.mi.fu-berlin.de/rojas/neural/chapter/K7.pdf

As Redes *Multilayer Perceptron* (RMP) possuem múltiplas camadas intermediárias (escondidas) e conseguem tratar de problemas que não sejam linearmente separáveis. Em uma rede multicamadas o processamento feito por cada neurônio depende dos resultados obtidos pelos neurônios da camada anterior. O número de neurônios nas camadas escondidas também influem nas respostas da rede, já que quanto maior o número de neurônios maior o numero de ligações.

A Figura 9 mostra um exemplo de RNA com três neurônios na camada de entrada, quatro na camada intermediária e dois na camada de saída.

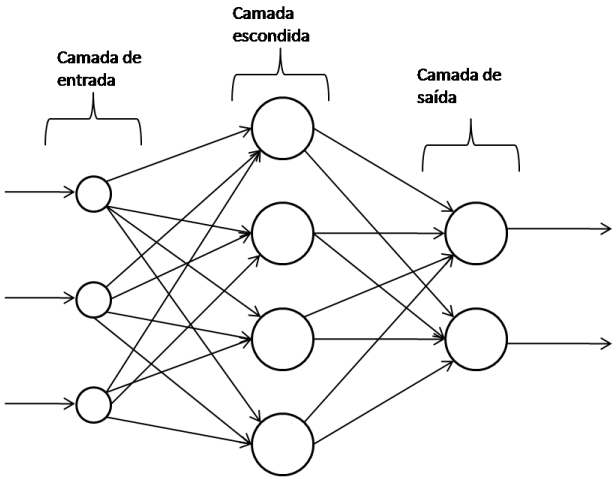


Figura 9 RNA com três camadas

3 METODOLOGIA

Este trabalho foi desenvolvido seguindo uma seqüência de etapas. Cada uma dessas etapas representa uma atividade do trabalho e elas foram executadas como é mostrado na Figura 10.

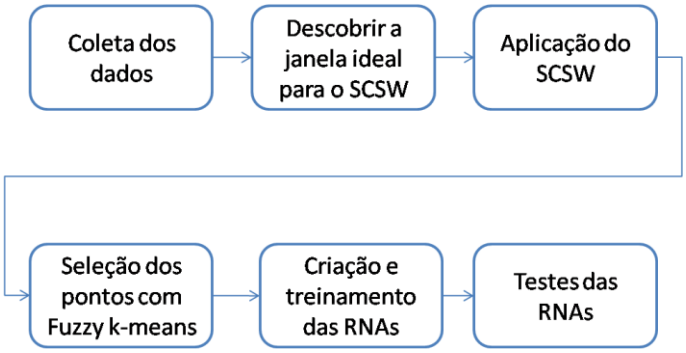


Figura 10 Etapas para a execução deste trabalho

3.1 Tipo de pesquisa

Esta pesquisa pode ser vista como tecnológica, pois visa à criação de um novo classificador de proteínas. Além disso, ela também pode ser considerada como uma pesquisa exploratória, experimental e em laboratório, de acordo com seus objetivos, procedimentos e local de execução respectivamente.

3.2 Formação da base de dados

Para a criação do classificador proposto no trabalho foi necessário a obtenção de uma base de dados inicial, que é utilizada no método de descoberta da janela ideal para a aplicação do SCSW (que será discutido na seção 3.2) e

também para o treinamento e validação das RNAs (que serão discutidos na seção 3.5).

A base de dados escolhida foi a *Clusters of Orthologous Groups* (COGs). Cada COG foi delineado comparando seqüências de proteínas codificadas em 43 genomas completos, representando 30 grandes linhagens filogenéticas (os dados desta pesquisa foram colhidos em abril de 2008).

Mais de 166 mil seqüências de proteínas foram extraídas dos COGs em arquivos FASTA e divididas de acordo com sua classificação funcional. Com isso foram criados 24 *clusters*, cada um representando uma classe funcional.

3.2.1 Transformação dos dados

As proteínas utilizadas em aplicações de bioinformática normalmente são representadas pelos aminoácidos que as formam. Isso se deve ao fato de que os bancos de dados públicos disponibilizam as seqüências de aminoácidos dessas proteínas.

O alfabeto que representa os diferentes tipos de aminoácidos nas proteínas normalmente conta com 20 caracteres diferentes. O maior problema de um alfabeto deste tamanho é que ele pode tornar o custo computacional de determinados métodos (como o SCSW) muito alto, dessa maneira é desejável que outros alfabetos sejam utilizados, como estratégia para redução de custo computacional.

Os aminoácidos possuem muitas propriedades, o que torna possível diferentes tipos de agrupamentos. O agrupamento utilizado neste trabalho é o *Exchange-group*, que foi utilizado por Wu *et al.* (1992) e é baseado na matriz de similaridade PAM (Dayhoff *et al.*, 1978), que leva em conta a probabilidade de

um aminoácido se transformar em outro (através de mutação). Este agrupamento junta os 20 tipos de aminoácidos em seis grupos possíveis, como visto no Quadro 4.

Quadro 4 Agrupamento dos 20 aminoácidos de acordo com o *Exchange-group*

H, R, K
D, E, N, Q
C
S, T, P, A, G
M, I, L, V
F, Y, W

Um exemplo dessa transformação pode ser vista na Figura 11, que mostra a transformação de uma seqüência com alfabeto de 20 caracteres (Figura 11(a)) em uma seqüência com alfabeto reduzido (Figura 11(b)), utilizando o *Exchange-group*.

H Y C H F E W G C M Q V
(a)

H F C H F H F S C M D M
(b)

Figura 11 Transformação de uma seqüência utilizando o alfabeto de 20 caracteres (a) em uma seqüência utilizando o alfabeto de seis caracteres (b) de acordo com o *Exchange-group*

Seguindo essa metodologia, todas as seqüências utilizadas durante este trabalho foram transformadas utilizando o *Exchange-group.*

3.3 SCSW

Para utilizar a codificação SCSW primeiro deve-se definir o tamanho ideal para a janela deslizante utilizada no método. Essa janela deve ser definida levando em conta tanto a confiabilidade dos vetores criados quanto o custo computacional.

Com a janela definida, o SCSW é aplicado em todo o conjunto que passar pelo classificador, na base de dados de treinamento, na de validação e em qualquer seqüência que for ser classificada pelo classificador, lembrando que as novas seqüências a ser classificadas serão previamente transformadas de acordo com o *Exchange-Group* como visto na seção 3.1.

3.3.1 Escolha do Tamanho de Janela Ideal

Para a escolha da janela ideal, todo o conjunto de dados é levado em conta, tanto os dados de treinamento quanto os de validação. A metodologia proposta em Pevzner *et al.* (1995), e descrita na seção 2.2 mostra os possíveis casos em que um vetor gerado a partir de uma seqüência pode apresentar ambigüidade.

Uma maneira para obter resultados a partir do método proposto por Pevzner *et al.*, (1995) pode ser vista na Figura 3.3.

1. Crie uma janela de tamanho *n*, onde *n* = 2.
2. Execute a metodologia com a janela de tamanho *n* em todas as seqüências da base de dados e armazene os resultados.
3. Caso alguma seqüência se encaixe em algum dos três casos, incremente o valor de *n* e volte ao passo 2. Caso contrário vá para o passo 4.
4. Com todos os dados de todas as execuções em mãos, calculamos a porcentagem de seqüências que se encaixam em pelo menos um dos 3 casos para um determinado tamanho de janela.

Figura 12 Algoritmo para a aplicação da metodologia de Pevzner *et al.* (1995)

Ao final da execução do algoritmo descrito na Figura 12 podemos inferir sobre a melhor escolha para um tamanho de janela. Neste trabalho a inferência foi realizada de acordo com a quantidade de seqüências que se encaixam em pelo menos um dos casos para uma determinada janela. Também foi levado em conta também o custo computacional que uma janela pode gerar, por exemplo, uma janela de tamanho $n = 5$ com um alfabeto a_m de tamanho $m = 6$ gera vetores de tamanho 7776.

O método descrito acima foi implementado e aplicado a todas as seqüências da base de dados, sendo que em alguns casos janelas de tamanho superior a 500 foram necessárias para que nenhuma seqüência entrasse em nenhum caso, e os resultados obtidos com este método podem ser vistos na Tabela 1.

Tabela 1 Resultados obtidos a partir da metodologia de Pevzner *et al.* (1995) com diferentes tamanhos de janela

Janela	**2**	**3**	**4**	**5**	**6**	**7**
Redundância	100,00%	99,97%	98,87%	90,69%	68,65%	36,08%

Tabela 1, conclusão

Janela	**8**	**9**	**10**	**11**	**12**	**13**
Redundância	13,35%	3,98%	1,43%	0,78%	0,54%	0,41%
Janela	**14**	**15**	**16**	**17**	**18**	**19**
Redundância	0,35%	0,29%	0,23%	0,21%	0,19%	0,17%

A Tabela 1 mostra como a possibilidade de redundância no SCSW diminuiu de acordo com o aumento do tamanho da janela. Com uma janela de tamanho dois a probabilidade de redundância no SCSW é de 100%, já com uma janela de tamanho 11 essa probabilidade não chega a 1%.

3.3.2 Aplicação do SCSW

Considerando os resultados obtidos e mostrados na Tabela 3.2, a melhor escolha para a aplicação do SCSW provavelmente seria a janela de tamanho sete (considerando o custo-benefício), mas essa escolha ainda é muito cara computacionalmente, pois iria gerar vetores de tamanho 6^7 (279.936 posições), o que é inviável, pois são mais de 166 mil seqüências diferentes. Se considerarmos que um número inteiro ocupe 16 *bits* de memória todas essas seqüências iriam ocupar no mínimo 86 GB de espaço em disco.

Tendo em vista a viabilidade do tamanho da janela, a janela escolhida foi a de tamanho cinco, que apesar de não ser a ideal ela foi a janela de maior tamanho que é viável para este trabalho, pois gera vetores de tamanho 6^5 (7.776 posições), que, apesar de utilizadas, ainda ocupam mais de 2,4 GB de espaço em disco no computador.

Com a janela de tamanho cinco, todas as seqüências da base de dados foram codificadas com o SCSW.

3.4 Estratégia para seleção de pontos com o algoritmo Fuzzy k-means

Apesar de ter uma base formada, o número de seqüências da base de dados é grande demais para a criação e treinamento de uma rede neural artificial, não havendo nem memória nem tempo hábil suficiente para treinamento de uma RNA com esta quantidade de dados. Desta maneira se fez necessária a utilização de alguma estratégia para seleção dos pontos que se encontram mais perto das bordas dos *clusters*, pois assim apenas estes pontos serão utilizados para a criação da RNA.

O algoritmo *Fuzzy k-means* (ou *Fuzzy c-means*) tem como função básica indicar a probabilidade de algum ponto pertencer a determinado cluster. Neste trabalho foi utilizada uma versão adaptada do algoritmo mostrado por Bezdek (1981).

A adaptação do algoritmo *Fuzzy k-means* implementada pelo autor deste trabalho consiste em:

1. Dividir cada *cluster* em dois *sub-clusters*.
2. Após a divisão, o algoritmo descrito na seção 2.2 (Figura 2.3) é executado para todos os pontos de cada *cluster* e eles são classificados de acordo com sua probabilidade de pertencer a um dos *sub-clusters*.
3. Selecionar os pontos que tenham maior probabilidade de pertencer a um dos *clusters*, mas que ao mesmo tempo estejam mais distantes dos centróides dos *sub-clusters*.

4. Caso a quantidade de pontos selecionada não seja satisfatória, selecione os pontos que estejam entre os dois *clusters*, mas que sejam mais distantes dos centróides.

É indesejado a utilização de pontos que estejam na divisa entre os *sub-clusters*, pois existe a possibilidade que eles não estejam nas bordas do *cluster* e sim apenas na divisa. Apesar de também escolher pontos que não estão nas bordas, o método apresentou bons resultados, e um exemplo de sua utilização pode ser visto na Figura 13.

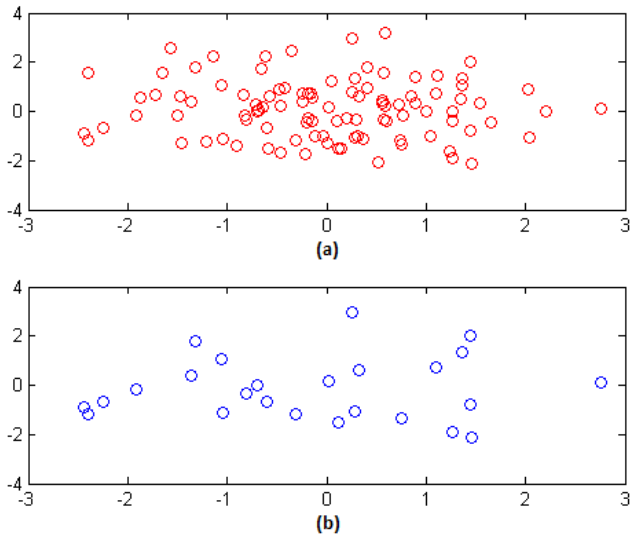


Figura 13 Aplicação do Fuzzy k-means implementado em um *cluster* qualquer

A Figura 13 mostra a aplicação do Fuzzy k-means implementado pelo autor. Na Figura 13(a) temos o *cluster* completo, com 100 pontos, na Figura 13(b) podemos ver os 25 pontos selecionados pelo algoritmo a partir do *cluster* inicial.

O número de pontos selecionados em cada *cluster* vai depender da utilização do método. Neste caso temos duas possibilidades, a primeira é escolher uma porcentagem de pontos do cluster e a segunda é escolher um número pré-fixado de pontos. Enquanto a primeira possibilidade se mostra a mais adequada, tendo em vista que o número de pontos selecionados será proporcional ao tamanho de cada *cluster*, a segunda se mostra uma boa opção no quesito custo computacional, uma vez que existem *clusters* com mais de 22 mil seqüências, ocupando mais de 330 MB de espaço em disco.

Tendo como objetivo a minimização do custo computacional, a estratégia utilizada foi a de escolher um número pré-definido de pontos em cada cluster.

Quadro 5 Número de proteínas em cada cluster formado

Cluster	Número de Proteínas	Cluster	Número de Proteínas
A	314	**M**	7858
B	547	**N**	2738
C	9830	**O**	6202
D	1677	**P**	9232
E	14938	**Q**	4055
F	3922	**R**	22721
G	10816	**S**	13888
H	6582	**T**	7681
I	5201	**U**	3737
J	10573	**V**	2380
K	11266	**W**	135
L	10339	**Z**	146

No Quadro 5 podemos ver o número de proteínas em cada um dos *clusters* do COG. O FCM selecionou 5% das seqüências de cada cluster para a criação dos conjuntos de treinamento e validação da RNA. Este número foi

escolhido tentando manter a proporção com o tamanho de cada conjunto, o grande problema é que essa proporção ainda é baixa, mas um conjunto maior teria um custo computacional muito alto para a RNA.

Após a seleção dos pontos, o conjunto de dados para a criação da rede foi formado contendo 24 *clusters*. O tamanho de cada *cluster* variou de acordo com o tamanho do conjunto original, variando entre seis pontos para a classe W e 1136 pontos para a classe R. Em cada conjunto 75% dos pontos foram escolhidos aleatoriamente para o treinamento da rede, e os 25% restantes formaram o conjunto de validação da RNA.

3.5 Criação e validação das Redes Neurais Artificiais

A RNA foi criada utilizando o *Neural Network ToolBox* do Matlab[11] 7.6.0.

A camada de entrada foi construída com 7776 neurônios, que é o tamanho da saída gerada pelo SCSW como visto na seção 3.2. A camada escondida foi criada com 10 neurônios, e a camada de saída com 24 neurônios. Essa topologia pode ser vista na Figura 14 (apesar de que os neurônios da camada escondida não estão sendo exibidos).

[11] http://www.mathworks.com

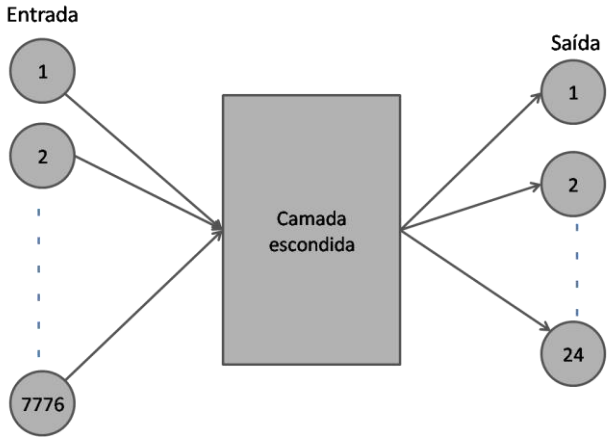


Figura 14 Topologia da RNA criada.

O treinamento dos neurônios foi feito utilizando a heurística *Resilient Backpropagation* durante 20 mil épocas de treinamento (repetições).

Ao final do treinamento, o conjunto de validação passou pela RNA e a saída da rede foi comparada com os resultados esperados para obtenção do índice de acerto da RNA.

4 RESULTADOS

Após o treinamento e validação da RNA, foram obtidos os resultados do classificador criado.

A Tabela 2 analisa a saída do classificador de duas maneiras:

- Acerto: mostra a porcentagem de pontos desta classe que foram classificados corretamente pela RNA. Essa porcentagem é calculada em relação ao número de pontos de validação. Logo se a classe V obteve 30 % de acerto, significa dizer que a RNA acertou nove dos 30 pontos selecionados para a validação.
- Erro: mostra a porcentagem de pontos que foram classificados de maneira incorreta pela rede. O cálculo é feito tomando por base a quantidade de pontos classificados incorretamente pela RNA dividido pela quantidade total de pontos classificados como sendo de determinada classe.

Tabela 2 Resultados obtidos com a RNA

	Classes Funcionais					
	A	**B**	**C**	**D**	**E**	**F**
Acerto	0,00%	0,00%	24,39%	9,52%	20,86%	14,29%
Erro	-	100,00%	63,41%	92,00%	71,32%	69,57%
	G	**H**	**I**	**J**	**K**	**L**
Acerto	20,74%	14,46%	20,00%	29,55%	30,50%	18,60%
Erro	79,10%	85,88%	81,43%	29,09%	77,84%	81,54%
	M	**N**	**O**	**P**	**Q**	**R**
Acerto	31,63%	41,18%	37,18%	18,97%	11,76%	29,58%
Erro	61,73%	75,44%	71,84%	82,95%	82,86%	78,41%
	S	**T**	**U**	**V**	**W**	**Z**
Acerto	26,44%	9,38%	14,89%	30,00%	100,00%	0,00%
Erro	74,86%	91,59%	74,07%	72,73%	50,00%	100,00%

Os resultados obtidos e apresentados na Tabela 2 mostram que a rede teve um índice de acertos baixo.

Um dos fatores que podem ter interferido nos resultados é o número de pontos selecionados pelo FCM. Em alguns casos, como na classe W, apenas quatro dos seus 135 pontos foram utilizados para o treinamento da RNA, o que pode tornar uma amostra tão pequena pouco útil no treinamento da rede.

Ao mesmo tempo, se outro tipo de amostragem fosse feito, como, por exemplo, a seleção de um número fixo de pontos de cada classe, a quantidade de pontos ao final ainda seria desbalanceada. Se, por exemplo, 135 pontos de cada conjunto fossem selecionados (que é o número de pontos da classe W, a menor delas), a classe R teria apenas 135 pontos para representar os seus mais de 22 mil pontos, o que é menos de 1% da classe. Ao passo que a classe W teria 100% dos seus pontos selecionados.

4.1 Resultados Específicos

Apesar dos resultados apresentados pela RNA não terem sido satisfatórios, este trabalho conseguiu atingir seus objetivos específicos, como mostrados na seção 1.2.2.

Uma base de dados foi construída a partir dos COGs, essa base de dados foi obtida e dividida de acordo com cada classe funcional. Além disso, essa base foi codificada de acordo com o *Exchange-group* (seção 3.2.1).

A metodologia para a descoberta da janela ideal para a aplicação do SCSW foi implementada e aplicada, mostrando quais são os tamanhos ideais de

janela para os dados. Os resultados obtidos puderam ser vistos na seção 3.3 (Tabela 1).

O FCM se mostrou uma estratégia interessante para a seleção dos pontos, e o modo utilizado, como método para seleção de pontos nas bordas dentro de um único cluster, como visto em Coelho *et al.* (2010), é uma abordagem diferente da normalmente empregada a algoritmos deste tipo. Um exemplo do resultado deste método pôde ser visto na seção 3.4 (Figura 13).

5 CONCLUSÕES

Os resultados obtidos com o classificador não foram boms, mas isso não significa que os métodos para classificação de proteínas com RNA possam ser descartados. Outros testes, como diferentes algoritmos de treinamento para os neurônios, mudança no número de épocas de treinamento ou mudança no número de neurônios na camada escondida precisam ser feitos para que novos resultados possam ser obtidos.

A seleção dos dados para treinamento da RNA é um ponto que deve ser levado em consideração, uma vez que o conjunto gerado para treinamento não é ideal para todas as classes. Uma nova seleção deve ser feita com cuidado, de maneira que o conjunto criado seja fiel ao conjunto original e factível para a criação da rede.

O método SCSW para codificação das seqüências se mostrou uma alternativa interessante, mas ele se torna computacionalmente caro quando janelas de maior ordem são utilizadas.

5.1 Trabalhos Futuros

Este trabalho pode gerar novos frutos, sua metodologia pode ser aproveitada e alguns novos métodos podem ser testados, como:

- Utilização de outros métodos para codificação das proteínas, como o *Extended-Sequence Coding by Sliding Window* proposto por Rodrigues (2007).

- Utilização de diferentes alfabetos, além do alfabeto de tamanho 6, ainda existem diferentes alfabetos que podem ser utilizados, como o de tamanho 20, o de tamanho 2 e o de tamanho 3.
- Outros métodos para seleção dos dados de criação e treinamento das RNAs, além do FCM existem diversos outros métodos para classificação de dados.
- Uso de outros algoritmos de treinamento para os neurônios da RNA.

6 REFERÊNCIAS BIBLIOGRÁFICAS

ALTSCHUL, S.F., GISH, W., MILLER, W. **Basic Local Alignment Search Tool.** Journal of Molecular Biology, 215, p. 403-410, 1990.

BENSON, D. A., KARSCH-MIZRACHI, I., LIPMAN, D. J. **GenBank**. Nucleic Acids Research, Vol. 37, 2009.

BEZDEK, J. C. **Pattern recognition with fuzzy objective function algorithms.** Plenum Press, New York, 1981.

BLAISDELL, B. E. **A measure of the similarity of sets of sequences not requiring sequence alignment**. Proc. Natl. Acad. Sci. USA, 83, (1986).

BRAGA, A. P., LUDERMIR, T. B., CARVALHO, A.C. **Redes Neurais Artificiais: Teoria e aplicações.** 1 ed., Livros Técnicos e Científicos - LTC, 2000.

COELHO, F., BRAGA, A. P., NATOWICZ, R. **Semi-supervised model applied to the prediction of the response to preoperative chemotherapy for breast cancer.** Soft Computing - A Fusion of Foundations, Methodologies and Applications, 2010.

COELHO, T. A. **Classificação de Proteínas com Redes Neurais Artificiais.** Universidade Federal de Lavras, 2008.

DAYHOF, M. O., SCHWARTZ R.M., ORCUTT, B.C. **A Model of Evolutionary Change in Proteins.** Atlas Of Protein Sequence And Structure, c. 22, p. 345-352, 1978.

KARRER, D., CAMEIRA, R., VASQUES, A. **Redes Neurais Artificiais: Conceitos e Aplicações.** IX Profundão - Encontro de Engenharia de Produção da UFRJ, 2005.

MATTEUCCI, M. **Fuzzy C-Means Clustering.** A Tutorial on Clustering Algorithms. Disponível em: http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/, acessado em 23 de maio de 2010.

NCBI, **Clusthers of Orthologous Groups**, National Center for Biotechnology Information. Disponível em: http://www.ncbi.nlm.nih.gov/COG/, acessado em: 23 de maio de 2010.

PEVZNER, P. A. **Dna physical mapping and alternating eulerian cycles in colored graphs**. Algorithmica, p. 77–105, 1995.

RODRIGUES, T. S. **Codificação de Sequências de Aminoácidos e sua Aplicação na Classificação de Proteínas com Redes Neurais Artificiais**, Universidade Federal de Minas Gerais, 2007.

RODRIGUES, T. S., BRAGA, A. P., PACÍFICO, L. G. **Amino acid coding with sliding window technique**. Procedings of Workshop of Bioinformatics, 2003.

WU, C., WHITSON, G., McLARTY, J. **Protein classification artificial neural system.** Protein Science, p. 667-677, 1992.