UNIVERSIDADE ESTADUAL DE LONDRINA

PAULO ROBERTO SILLA

IDENTIFICAÇÃO DE miRNAs E SEUS ALVOS NO GENOMA DA SOJA COM O USO DE ABORDAGENS COMPUTACIONAIS

Londrina
2011

PAULO ROBERTO SILLA

IDENTIFICAÇÃO DE miRNAs E SEUS ALVOS NO GENOMA DA SOJA COM O USO DE ABORDAGENS COMPUTACIONAIS

Trabalho de Dissertação de Mestrado apresentado á Universidade Estadual de Londrina como parte dos requisitos para obtenção do titulo de Mestre em Ciência da Computação.

Orientadora: Profa. Dra. Maria Angélica de Oliveira Camargo-Brunetto.
Co-orientador: Dr. Eliseu Binneck.

Londrina
2011

Catalogação elaborada pela Divisão de Processos Técnicos da Biblioteca Central da Universidade Estadual de Londrina

Dados Internacionais de Catalogação-na-Publicação (CIP)

S584i Silla, Paulo Roberto
Identificação de miRNAs e seus alvos no genoma da soja com o uso de abordagens computacionais / Paulo Roberto Silla. – Londrina, 2011
85 f.: il.

Orientador: Maria Angélica de Oliveira Camargo-Brunetto.
Co-orientador: Eliseu Binneck.
Dissertação (Mestrado em Ciência da Computação) - Universidade Estadual de Londrina, Centro de Ciências Exatas, Programa de Pós-Graduação em Ciência da Computação, 2011.
Inclui bibliografia.

1. Inteligência Artificial – Teses. 2. Inteligência Artificial – Aplicações Biológicas – Teses. 3. Homologia (Biologia) – Teses. 4. Soja – Ácido Ribonucléico – Teses. 5. Bioinformática – Teses. I. Camargo-Brunetto, Maria Angélica de Oliveira. II. Binneck, Eliseu. III. Universidade Estadual de Londrina. Centro de Ciências Exatas. Programa de Pós-Graduação em Ciência da Computação. IV. Título.

CDU 519.683

PAULO ROBERTO SILLA

IDENTIFICAÇÃO DE miRNAs E SEUS ALVOS NO GENOMA DA SOJA COM O USO DE ABORDAGENS COMPUTACIONAIS

Trabalho de Dissertação de Mestrado apresentado á Universidade Estadual de Londrina como parte dos requisitos para obtenção do titulo de Mestre em Ciência da Computação.

BANCA EXAMINADORA

_____

Profa. Dra. Maria Angélica de Oliveira Camargo-Brunetto
Universidade Estadual de Londrina - UEL

_____

Prof. Dr. Pedro Paulo da Silva Ayrosa
Universidade Estadual de Londrina - UEL

_____

Prof. Dr. Wesley Attrot
Universidade Estadual de Londrina - UEL

_____

Prof. Dr. Rodrigo Matheus Pereira
Universidade Federal da Grande Dourados - UFGD

Londrina, 20 de Junho de 2011.

Agradecimentos

Agradeço a Deus pela vida;

A minha orientadora Prof[a]. Dra. Maria Angélica pelos ensinamentos e por sua dedicação;

A meu co-orientador Dr. Eliseu Binneck pela sugestão do tema e pela oferta do estagio na Embrapa;

Aos meus pais Roberto e Lourdes por sempre acreditarem em mim;

Aos meus irmãos Marcos e Mateus e ao restante da minha família pelo apoio;

A minha noiva Elaine por estar sempre ao meu lado;

Aos colegas do Mestrado em Ciência da Computação da UEL e do Laboratório de Biotecnologia

Vegetal e Bioinformática da Embrapa Soja pelos momentos compartilhados;

Ao CNPq pelo apoio financeiro.

À memória de Mario Silla

"A persistência é o menor caminho do êxito."

Charles Chaplin

SILLA, Paulo R. **Identificação de miRNAs e seus alvos no genoma da soja com o uso de abordagens computacionais**. 2011. 85 f. Dissertação (Mestrado) - Universidade Estadual de Londrina, Londrina. 2011.

RESUMO

MicroRNAs (miRNAs) são pequenas moléculas de ácido ribonucléico que não codificam proteína (*non-coding ribonucleic acid - ncRNA*), descobertas no início dos anos 1990, presentes em animais, plantas e vírus, as quais acreditava-se não possuir função no organismo. Porém, recentemente, descobriu-se que os miRNAs possuem papel regulatório na expressão gênica a nível pós-transcricional, fato que despertou grande interesse na comunidade científica. Na literatura, os trabalhos iniciais relacionados à identificação de miRNAs utilizavam técnicas experimentais. Porém, devido à grande quantidade de dados disponíveis para análise, abordagens computacionais começaram a ser utilizadas, destacando-se a busca por homologia e o uso de aprendizagem de máquina (técnica conhecida como *ab initio*). Em relação à identificação de miRNAs na soja, a literatura apresenta o uso de técnicas experimentais e da busca por homologia. Neste trabalho, foram identificadas na soja (*Glycine max*), três sequencias de pre-miRNAs de outras plantas através de buscas por homologia. Também foi desenvolvido um classificador com o uso do algoritmo de Máquina de Vetor de Suporte (*Support Vector Machine* - SVM) voltado para a predição de miRNAs em plantas e otimizado com técnicas de computação paralela. Este classificador foi utilizado a fim de investigar a existência de pre-miRNAs em 1205 sequências retiradas dos íntrons da soja, resultando em 160 possíveis sequências de pre-miRNA.

Palavras-Chave: Soja (*Glycine max*), Bioinformática, Ácido Ribonucléico, Máquina de Vetor de Suporte, Computação paralela.

SILLA, Paulo R. **Identification of miRNAs and their targets in the soybean genome using computational approaches**. 2011. 85 p. Dissertation (Master's degree) - State University of Londrina, Londrina. 2011.

ABSTRACT

MicroRNAs (miRNAs) are small ribonucleic acid molecules that do not encode proteins (non-coding ribonucleic acid - ncRNA), discovered in early 1990. MiRNAs are present in animals, plants and viruses, but it was believed that they had no role in organism. However recently, it was discovered that miRNAs have regulatory function in gene expression at post-transcriptional level, a fact which attracted great interest in the scientific community. The initial works related to the identification of miRNAs used experimental techniques in this task. But due to the large amount of data available for analysis, computational approaches began to be used, emphasizing the search for homology and the use of machine learning (a technique known as *ab initio*). In relation the identification of miRNAs in soybean, the literature presents the use of experimental techniques and the search for homology. In this work, have been identified in soybean (*Glycine max*), three pre-miRNAs sequences from other plants using the search for homology. Also, a pre-miRNA classifier was developed using the algorithm of Support Vector Machine - SVM, directed to the prediction of miRNAs in plants and optimized with parallel computing techniques. This classifier was used to investigate the existence of pre-miRNAs in 1205 sequences extracted from soybean introns, resulting in 160 possible pre-miRNA sequences.

Keywords: Soybean (*Glycine max*), Bioinformatics, Ribonucleic Acid, Support Vector Machine, Parallel Computing.

Lista de Figuras

Lista de Tabelas

Lista de Siglas e Abreviaturas

ADN	Acido Desoxirribonucleico
AMFE	Adjusted Minimal Folding Free Energy
ARN	Acido Ribonucléico
BLAST	Basic Local Alignment Search Tool
CPU	Central Processing Unit
CT	Conjunto de Treinamento
CV	Conjunto de Validação
DNA	Deoxyrribonucleic acid
Espec.	Especificidade
Exat.	Exatidão
FN	Falsos Negativos
FP	Falsos Positivos
GB	Gigabytes
MFE	Minimal Folding Free Energy
MFEI	Minimal Folding Free Energy Index
miRISC	Complexo de Silenciamento Induzido por miRNA
miRNA	MicroRNA
mRNA	RNA mensageiro
NCBI	National Center for Biotechnology Information
ncRNA	RNA não codificante
nt(s)	Nucleotídeo(s)
pre-miRNA	miRNA precursor
pri-miRNA	miRNA primário
RAM	Random Access Memory
RNA	Ribonucleic acid
rRNA	RNA ribossômico
Sens.	Sensibilidade
SOAP	Short Oligonucleotide Alignment Program
SVM	Support Vector Machine
tRNA	RNA de transferência
VN	Verdadeiros Negativos
VP	Verdadeiros Positivos

Sumário

1 INTRODUÇÃO

A soja é uma das culturas agrícolas mais importantes do mundo, devido ao seu papel na indústria alimentícia e na produção de biocombustíveis. O Brasil é o segundo maior produtor mundial dessa leguminosa, com produção de $68,69$ milhões de toneladas na safra 2009/2010, usando área de $23,24$ milhões de hectares (Conab 2010). Devido a desafios como a diversidade de clima e solo existentes no Brasil, a ocorrência de pragas e doenças e, principalmente, a preservação de áreas florestais, existe uma clara necessidade de que sejam desenvolvidas cultivares de soja melhor adaptadas e, consequentemente, mais produtivas.

No último século, muitas informações a respeito da estrutura molecular dos organismos vivos foram decifradas e, nas últimas décadas, a velocidade e facilidade de identificação dos constituintes moleculares foi facilitada pelo sequenciamento do genoma de vários organismos, de modo que muitos processos que ocorrem nas células foram caracterizados (Bruggeman et al. 2007).

Um dos importantes avanços recentes da pesquisa biológica foi a descoberta de pequenas moléculas de ácido ribonucleico (ARN) ou *ribonucleic acid (RNA*[1]*)* versáteis que possuem em torno de 18 a 22 nucleotídeos e controlam a expressão gênica em nível pós-transcricional (Ghosh, Chakrabarti e Mallick 2007) (Sunkar e Jagadeeswaran 2008). Tais moléculas, chamadas microRNA (miRNA), regulam de modo negativo o produto de seu gene alvo (pela clivagem do RNA mensageiro e posterior degradação ou atenuação da tradução, impedindo a produção da proteína íntegra no processo de expressão do gene alvo).

A tarefa de descobrir miRNAs presentes nos organismos e determinar quais genes são regulados por eles apresenta-se como parte fundamental para o entendimento do mecanismo de muitas doenças em animais e plantas. Na soja, sabe-se que os miRNAs executam papel importante para seu desenvolvimento, fixação de nitrogênio e atuam em resposta a estresses bióticos e abióticos (Zhang, Pan e Stellwag 2008).

As metodologias usadas para identificação de miRNAs podem ser divididas em dois grupos (Hou, Ying e Li 2008): *i)* métodos experimentais baseados em clonagem e sequenciamento de pequenos RNAs *(small RNA - sRNA)* – abordagem limitada, pois alguns

[1]No restante deste trabalho, será utilizada a sigla RNA para referir-se ao ácido ribonucleico.

miRNAs são expressos em níveis muito baixos e/ou expressos especificamente em determinados tecidos e num determinado momento do desenvolvimento do organismo, o que dificulta a amostragem exaustiva de todos os miRNAs; *ii)* métodos computacionais – abordagem que tem o potencial de predizer todos os miRNAs, inclusive os pouco abundantes e expressos em tecidos específicos e, por isso, tais métodos vêm sendo utilizados em complemento às técnicas experimentais.

As duas principais abordagens computacionais utilizadas para a identificação de miRNAs são:

- Busca baseada em homologia (Mendes, Freitas e Sagot 2009): comparação de sequências candidatas contra miRNAs conhecidos. Com este método, podem ser encontrados miRNAs conservados entre espécies. Em plantas, porém, a maioria dos genes miRNA são não conservados, o que limita a utilização dessa abordagem (Zhang et al. 2005). Um outro fator negativo da busca baseada em homologia é que ela não identifica novos miRNAs.

- Aprendizagem de máquina (*ab initio*): aplicados mais recentemente na descoberta de miRNA. Métodos derivados de aprendizagem de máquina têm apresentado bons resultados (Yousef, Showe e Showe 2009), (Sewer et al. 2005), (Loong e Mishra 2007). Basicamente, esta técnica necessita de um conjunto formado por miRNAs conhecidos e sequências aleatórias para ser utilizado como parâmetro de treinamento. Após o treinamento, testes devem ser realizados a fim de medir a exatidão do algoritmo classificador.

  Finalmente, o conjunto onde deseja-se descobrir novos miRNAs (miRNAs candidatos) é apresentado ao algoritmo e este classifica cada sequência em uma das seguintes classes: miRNA ou não miRNA. Deste modo, o classificador indica quais sequências candidatas estão mais próximas das sequências miRNA conhecidas e, assim, determina aquelas que possuem maior probabilidade de serem verificadas como um miRNA verdadeiro.

1.1 Trabalhos Relacionados

Esta seção apresenta trabalhos relacionados a métodos computacionais para a identificação de miRNAs e está organizada de modo que as Subseções 1.1.1 e 1.1.2 apresentam trabalhos que utilizaram busca por homologia e a aprendizagem de máquina, respectivamente, em suas abordagens para identificação de miRNAs.

1.1.1 Identificação Baseada em Busca por Homologia

Nesta Subseção serão analisados os trabalhos de identificação de miRNAs baseados na busca por homologia desenvolvidos voltados para plantas.

O MIRFINDER, desenvolvido por (Bonnet et al. 2004) é um *pipeline* computacional para a identificação de miRNAs em *Arabidopsis thaliana*, uma planta da família da mostarda, através da comparação de seu genoma com o genoma da *Oryza sativa* (arroz).

No desenvolvimento do MIRFINDER, os autores consideraram as seguintes regras gerais: *i)* a sequência do miRNA é conservada entre a *Arabidopsis* e o arroz, mesmo que o restante da sequência precursora seja diferente; *ii)* a habilidade da sequência precursora formar grampo na estrutura secundária ser conservada em ambos os genomas comparados; e *iii)* para dois miRNAs ortólogos, a sequência está sempre localizada no mesmo braço do grampo, nas duas espécies.

Além das regras, os autores consideraram outros cinco parâmetros para selecionar estruturas de grampo mais significativas, descritos a seguir: *i)* o miRNA deve ser parte de uma hélice contínua; *ii)* o valor da mínima energia livre deve ser menor que -30 kcal/mol; *iii)* o número mínimo de resíduos pareados no miRNA deve ser 15; *iv)* o número máximo de resíduos que não formam par nas fitas do miRNA e na sua complementar deve ser cinco; e *v)* o número máximo de pares G·U presentes no miRNA deve ser cinco.

O MIRFINDER identificou 91 potenciais miRNAs presentes em *Arabidopsis thaliana*, dos quais observou-se que 58 possuem similaridade perfeita com pelo menos um RNA mensageiro presente no genoma desta planta.

No trabalho de Jones-Rhoades e Bartel 2004 os autores desenvolveram uma abordagem computacional a fim de identificar miRNAs em plantas, mais precisamente, miRNAs conservados entre *Arabidopsis thaliana* e *Oryza sativa*.

A busca foi baseada em seis características presentes em sequências de miRNA, conforme descrito a seguir: *i)* O pareamento de bases do miRNA maduro com o miRNA*[2] internamente ao grampo precursor é relativamente consistente. Em contrapartida, tanto o tamanho do *foldback* quanto a extensão do emparelhamento de bases fora da vizinhança imediata do miRNA são altamente variáveis entre os grampos de miRNA em plantas, mesmo nos miRNAs pertencentes a genes da mesma família. *ii)* A maioria dos miRNAs conhecidos em *Arabidopsis thaliana* possuem homólogos no genoma da *Oryza sativa*, com substituições de zero a duas bases; *iii)* A estrutura secundária em formato de grampo do miRNA conhecido é robustamente predita pelo *software* RNAfold quando dada uma sequência suficientemente longa para conter o miRNA maduro e o miRNA*; *iv)* As sequências dos grampos de *Arabi-*

[2]miRNA* (miRNA estrela) é a sequência complementar ao miRNA maduro. Mais detalhes podem ser obtidos na Subseção 2.1.1 desta dissertação, que trata da biogênese da molécula do miRNA.

dopsis thaliana e *Oryza sativa* são geralmente mais conservadas no miRNA e miRNA* que no segmento entre o miRNA e o miRNA*. *v)* Todos os acertos para miRNAs conhecidos no genoma da *Arabidopsis thaliana*, exceto para o anti-senso de regiões codificadoras, têm potencial para formar um grampo que pode conter um miRNA e, assim, são anotados como genes miRNA; *vi)* MiRNAs conhecidos de *Arabidopsis thaliana* são altamente complementares a um RNA mensageiro alvo, e esta complementariedade é conservada na *Oryza sativa*. Nesse trabalho, os autores encontraram em *Arabidopsis thaliana* 23 miRNAs conhecidos em *Oryza sativa*.

Em Zhang, Pan e Stellwag 2008 os autores realizam a predição de miRNAs em três espécies de soja: *Glycine max, Glycine soja* e *Glycine clandestine*, através de uma busca por homologia com o uso de sequências presentes no banco de dados público NCBI (`http://www.ncbi.nlm.nih.gov/`) com 184 miRNAs conhecidos da espécie *Arabidopsis thaliana*.

Neste trabalho, os autores consideraram um miRNA candidato se sua sequência: *i)* não possui mais que quatro substituições de nucleotídios se comparada ao miRNA de *Arabidopsis thaliana*; *ii)* encaixa-se em uma estrutura secundária apropriada (formato de grampo); *iii)* o miRNA maduro encontra-se em um braço do grampo; *iv)* não existe mais que seis diferenças de nucleotídios (*mismatches*) entre a sequência madura predita e sua sequência oposta miRNA* na estrutura secundária; *v)* não existe volta (*loop*) ou quebra nas sequências miRNA e miRNA*; *vi)* a estrutura secundária predita possui alto Índice de Energia Mínima Livre de Dobramento (Minimal Folding Free Energy Index - MFEI) e Energia Mínima Livre de Dobramento (*Minimal Folding Free Energy - MFE*) negativa.

Através desta técnica, os autores identificaram 69 miRNAs para a *Glycine max*, três para a *Glycine soja* e dois para a *Glycine clandestine*.

1.1.2 Identificação Baseada no Uso de Aprendizagem de Máquina

Abordagens desenvolvidas com o uso de aprendizagem de máquina para a identificação de miRNAs trabalham, geralmente, com a sequência do miRNA precursor (pre-miRNA), devido ao maior comprimento deste tipo de sequência, em relação ao comprimento da sequência do miRNA maduro. Este maior comprimento possibilita que seja estabelecido um número maior de atributos.

Ao realizar a identificação de sequências de pre-miRNA com o uso do algoritmo de Máquina de Vetor de Suporte (*Support Vector Machine - SVM*), Sewer et al. 2005 determinaram 40 características com base na sequência estrutural e de nucleotídeos, conforme descrito a seguir: *i)* 16 atributos estatísticos calculados sobre a estrutura completa do grampo; *ii)* 10 atributos estatísticos calculados sobre a região simétrica mais distante do caule;

iii) 11 atributos estatísticos calculados sobre a região mais distante na qual as diferenças entre os componentes 5' e 3'[3] das voltas assimétricas não são maiores que uma variação Δl; iv) três atributos estatísticos calculados sobre todas as janelas de tamanho correspondente ao do miRNA maduro que pode ser encontrado na sequência candidata.

Os autores utilizaram como heurística o fato de que os miRNAs ocorrem em grupos no genoma. O conjunto de candidatos a pre-miRNA foi composto por 3829 sequências originárias do genoma humano, 3537 do genoma do camundongo e 3034 do genoma do rato. Com o uso deste método foi determinada a predição de 224 pre-miRNAs no genoma humano, 192 em camundongo e 208 em rato. Após a remoção de sequências que foram preditas, mas já eram conhecidas, restaram 89, 66 e 105 novos pre-miRNAs nos genomas humano, do camundongo e do rato, respectivamente.

Xue et al. 2005 utilizam o algoritmo SVM com o propósito de desenvolver um método *ab initio*, chamado 3SVM, capaz de determinar sequências pre-miRNA verdadeiras a partir de um conjunto de sequências candidatas. Os atributos, elementos *triplet*, são obtidos sobre a representação estrutural da sequência, estabelecida através do programa RNAfold (Hofacker et al. 1994). Tal representação é feita por parêntesis, quando há o pareamento de bases ou por ponto, quando a base não faz par com outra.

O método *triplet* é composto de 32 características que são calculadas para cada nucleotídeo presente na sequência, observando seus nucleotídeos adjacentes. As características são formadas pelo nucleotídeo analisado, seguido por uma das composições observadas na representação da estrutura, que podem ser: "(((", "((.", "(..", "(.(", ".((", ".(.", "..(" e "...". O valor de cada característica é o número de vezes que ela ocorre na sequência. Finalmente os dados passam por um processo de normalização antes de sua utilização como entrada para o algoritmo SVM.

Os dados de treinamento e teste foram obtidos do genoma humano, sendo 193 sequências pre-miRNA verdadeiras, 8494 sequências que codificam proteína e 2444 grampos conservados entre o genoma humano e o genoma do rato. O algoritmo foi treinado com 163 pre-miRNAs humanos e 168 falsos pre-miRNA, selecionados aleatoriamente a partir dos pre-miRNAs verdadeiros e das sequências que codificam proteína, respectivamente.

Os testes com dados do genoma humano foram realizados com as 30 sequências pre-miRNA não utilizadas no treinamento e outras 39 novas sequências pre-miRNA, que foram reportadas durante o desenvolvimento do trabalho, totalizando 69 pre-miRNAs verdadeiros. Outros dois conjuntos de teste foram criados: o primeiro contendo todas as 2444 sequências de grampos conservadas e o segundo formado por 1000 sequências escolhidas aleatoriamente entre as sequências que codificam proteína e que não foram utilizadas no trei-

[3]Detalhes sobre biologia molecular podem ser obtidos na Seção 2.1 desta dissertação.

namento do algoritmo. A exatidão média dos resultados foi de $90,7\%$.

Testes com 581 sequências de pre-miRNA, pertencentes a organismos de 11 espécies distintas (entre elas duas espécies de plantas) também foram executados, e resultaram em $90,9\%$ de exatidão média.

O miPred, desenvolvido no trabalho de Loong e Mishra 2007, é um classificador SVM que utiliza a Função de Base Radial (*Radial Basis Function - RBF)* como função núcleo. Os autores determinaram 29 atributos de cada sequência e treinaram o algoritmo com 200 pre-miRNAs humanos e 400 sequências de falsos grampos. O miPred apresentou performance de predição igual ou superior a outros métodos existentes (ProMir (Nam et al. 2005), 3SVM (Xue et al. 2005), BayesMirFinder (Yousef et al. 2006), RNAmicro (Hertel e Stadler 2006) e (Sewer et al. 2005)) em relação à sensibilidade e especificidade, uma vez que foi capaz de identificar pre-miRNAs de diferentes espécies de organismos com alta exatidão. Por exemplo, em um conjunto de testes consistindo de 1918 pre-miRNAs e 3836 falsos grampos, o miPred apresentou sensibilidade, especificidade e exatidão iguais a 87.65%, 97.75% e 94.38% respectivamente.

Jiang et al. 2007 desenvolveram o MiPred (o nome diferencia-se do anterior apenas pela letra inicial maiúscula) que utiliza o algoritmo Floresta Aleatória (*Random Forest - RF*), combinado com um conjunto híbrido de atributos das sequências, o qual é formado pela composição local dos *triplets* estruturais contíguos, mínima energia livre de dobramento (MFE) e valores-P obtidos através de teste de aleatorização Monte Carlo. O processo de aprendizagem do algoritmo é realizado com dados de pre-miRNAs verdadeiros e falsos, ambos pertencentes ao genoma humano, onde é utilizado o mesmo conjunto de dados de treinamento do classificador Triplet-SVM, desenvolvido em (Xue et al. 2005).

Seu processo de classificação é executado em duas etapas: dada uma sequência, o MiPred determina se esta é um grampo capaz de formar pre-miRNA ou não. Em seguida, o algoritmo de Floresta Aleatória prediz se esta é um pre-miRNA verdadeiro ou falso. Os resultados de seus testes, também realizados com dados originados do genoma humano, apresentaram $93,21\%$ de especificidade, $89,35\%$ de sensibilidade e exatidão igual a $91,29\%$.

O preditor PMirP (Zhao et al. 2010) também baseia-se no método triplet-SVM (Xue et al. 2005) e utiliza o algoritmo SVM com a Função de Base Radial Gaussiana como núcleo. O conjunto de atributos apresentado ao classificador é composto por 32 características estruturais obtidas pelo método *left-triplet*, dois atributos calculados com base na mínima energia livre de dobramento (MFE) e dois atributos obtidos através do número de pares de base da sequência. Para o treinamento/teste do PMirP, foram utilizadas apenas sequências do genoma humano. Os resultados da predição de pre-miRNAs humanos, onde 69 sequências são pre-miRNAs verdadeiros e 3440 são falsos, apresentaram exatidão média de $95,4\%$ contra $90,7\%$ do método Triplet-SVM.

Quando aplicado a outras espécies, o PMirP apresentou 94% de exatidão, contra $90,9\%$ do método triplet-SVM em um total de 581 sequências pre-miRNA, das quais 80 pertencem a duas espécies de planta (*Oryza sativa* e *Arabidopsis thaliana*).

A Tabela 1.1 destaca os pontos principais dos trabalhos relacionados à identificação de pre-miRNAs baseada em aprendizagem de máquina.

Tabela 1.1: Características do trabalhos relacionados à identificação de pre-miRNAs baseada em aprendizagem de máquina.

Trabalho	**Algoritmo**	**Atributos**	**Genoma Treinamento**	**Genoma Teste**
Sewer et al. 2005.	SVM.	40 atributos estatísticos calculados sobre a sequência estrutural e de nucleotídeos.	Humano.	Novos pre-miRNAs: 89 humano, 66 camundongo, 105 rato.
3SVM (Xue et al. 2005).	SVM.	32 elementos *Triplet*.	Humano: 163 pre-miRNAs; 168 falsos pre-miRNAs.	Humano: $90,7\%$ de exatidão. Outras espécies: $90,9\%$ de exatidão.
miPred (Loong e Mishra 2007).	SVM.	29 atributos globais e intrínsecos das sequências.	Humano: 200 pre-miRNAs; 400 falsos pre-miRNAs.	Diversas espécies: $94,38\%$ de exatidão.
MiPred (Jiang et al. 2007).	Floresta Aleatória.	Elementos *Triplet*, MFE e valores-P.	Humano: 163 pre-miRNAs; 168 falsos pre-miRNAs.	Humano: $91,29\%$ de exatidão.
PMirP (Zhao et al. 2010).	SVM.	Elementos *Triplet*, MFE e número de pares de base.	Idem (Xue et al. 2005).	Humano: $95,4\%$ de exatidão. Outras espécies: 94% de exatidão.

1.2 Justificativa

A busca por homologia empregada neste trabalho tem por finalidade identificar na soja (*Glycine max*), pre-miRNAs de espécies de plantas evolucionariamente próximas

e que ainda não haviam sido descobertas em Zhang, Pan e Stellwag 2008.

Para o método baseado em aprendizagem de máquina, como os preditores *ab initio* desenvolvidos em (Xue et al. 2005), (Sewer et al. 2005), (Loong e Mishra 2007) e (Jiang et al. 2007), não foi encontrada, na literatura relacionada, sua aplicação para a predição de novos pre-miRNAs na soja. Deste modo, o preditor desenvolvido neste trabalho, está voltado para a identificação de novos pre-miRNAs em plantas, uma vez que seu treinamento foi realizado com o conjunto positivo formado por sequências pre-miRNA conhecidas de plantas.

1.3 Objetivos

Os objetivos deste trabalho são identificar pre-miRNAs na soja *Glycine max* pelo uso de técnicas computacionais, do seguinte modo: *i)* através da busca por homologia, na qual pre-miRNAs conhecidos de espécies de plantas evolucionariamente próximas à soja serão utilizados contra seu genoma e, *ii)* através de aprendizagem de máquina, com o desenvolvimento de um classificador SVM voltado para a predição de pre-miRNAs em plantas. As sequências candidatas a pre-miRNA apresentadas a este classificador serão obtidas de regiões intergênicas do genoma, de introns e das regiões 5'UTR e 3'UTR dos transcritos da soja.

1.4 Organização do Texto

Esta dissertação está organizada em seis capítulos, conforme a descrição a seguir: o Capítulo 1 apresenta a introdução ao tema, a revisão bibliográfica, os objetivos do trabalho e a organização do texto; o Capítulo 2 mostra os conceitos biológicos necessários para o entendimento do restante do texto; o Capítulo 3 introduz os conceitos computacionais estudados e utilizados no desenvolvimento do trabalho; o Capítulo 4 apresenta a metodologia empregada no desenvolvimento do trabalho; o Capítulo 5 apresenta o classificador resultante e também lista os possíveis pre-miRNAs identificados no genoma da soja, juntamente com seus alvos; o Capítulo 6 mostra as conclusões e os trabalhos futuros.

2 FUNDAMENTAÇÃO BIOLÓGICA

Neste capítulo estão descritos os conceitos básicos de biologia molecular necessários para o entendimento do restante do trabalho. O conceito fundamental mostrado aqui é conhecido como Dogma Central da Biologia Molecular (Seção 2.1), na qual a informação contida na sequência de ácido desoxirribonucleico (ADN) ou *deoxyrribonucleic acid (DNA*[1]*)* é traduzida em proteína. A Seção 2.1.1 descreve a biogênese dos miRNAs e o modo como estes interferem na expressão gênica dos organismos e a Seção 2.2 mostra uma breve introdução sobre a evolução de sequências.

2.1 Biologia Molecular

Célula é a menor unidade de vida contendo as características morfológicas e fisiológicas de todos os organismos vivos (Zaha et al. 1996). A classificação dos organismos, considerando seu nível celular é realizada em dois grupos: organismos procarióticos (unicelulares) e organismos eucarióticos[2] que, além de possuírem mais de uma célula, estas são mais complexas se comparadas às células dos organismos procarióticos.

As células possuem regiões definidas, separadas do citoplasma por membranas internas formando compartimentos chamados organelas, como pode ser observado na Figura 2.1, a qual apresenta uma célula vegetal.

Em cada organela se realizam funções especializadas (Zaha et al. 1996). No núcleo, por exemplo, estão presentes as moléculas de DNA.

A informação hereditária de todos os organismos vivos, com exceção de alguns vírus está contida no DNA (Graur e Li 2000), que é composto por nucleotídeos dispostos de forma sequencial em duas fitas. Cada nucleotídeo possui um grupo fosfato, um açúcar (pentose) e uma base. As bases são Adenina (A) e Guanina (G) (purinas) e Timina (T) e Citosina (C) (pirimidinas). A pentose presente no DNA é a desoxirribose.

[1]No restante do trabalho, será utilizada a sigla DNA para referir-se ao ácido desoxirribonucléico.

[2]Devido a este trabalho focar o estudo da soja, que é um organismo eucariótico, nas demais referências à célula serão consideradas apenas as células eucarióticas.

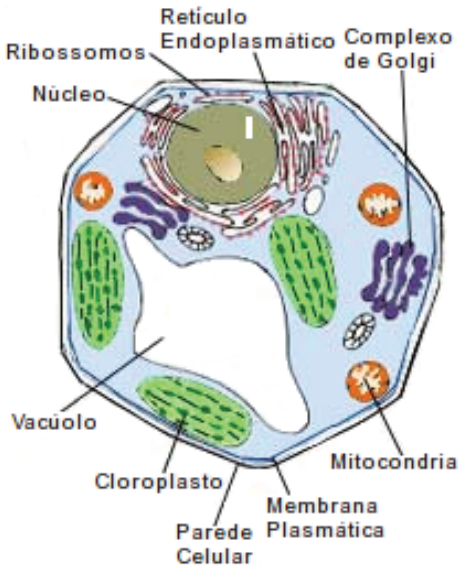

Figura 2.1: Célula eucariótica vegetal com suas organelas. As moléculas de DNA estão presentes no núcleo, onde ocorrem os processos de replicação e transcrição.

Em cada uma das fitas do DNA, os nucleotídeos unem-se através de ligações fosfodiéster. Essas ligações ocorrem do seguinte modo: o grupo hidroxila presente no carbono-3 da pentose do primeiro nucleotídeo liga-se ao grupo fosfato da hidroxila presente no carbono-5 da pentose do nucleotídeo seguinte, como pode ser observado na Figura 2.2, que mostra um fragmento de DNA com oito nucleotídeos (`TCGA` na fita principal e, consequentemente, `AGCT` na fita complementar).

As fitas do DNA são complementares de modo que uma purina sempre faz par com uma pirimidina, mais especificamente, Adenina liga-se com Timina e Citosina liga-se com Guanina, formando pares de bases canônicas através de pontes de hidrogênio (Graur e Li 2000), fato que atribui à molécula de DNA o formato de dupla hélice em torno de um eixo central (Zaha et al. 1996).

Por convenção, o crescimento das cadeias de nucleotídeos é representado na orientação $5' \rightarrow 3'$ (cinco linha para três linha), ficando o carbono-5 da pentose voltado para cima (Zaha et al. 1996).

Todos os organismos replicam seu DNA antes da divisão celular (Graur e Li 2000), como forma de garantir que as novas células carregarão a informação genética. Durante a replicação, cada fita é utilizada como modelo para criar sua imagem espelhada (Gibas e Jambeck 2001). As moléculas de DNA resultantes possuem uma fita do DNA que pertencia à célula mãe e uma fita nova (Graur e Li 2000).

O ácido ribonucleico (RNA) é sintetizado a partir do DNA através do processo conhecido como transcrição (Zaha et al. 1996) e sua formação consiste de um grupo fosfato, um açúcar e uma base. A estrutura do RNA é similar à do DNA, exceto por seu açúcar ser

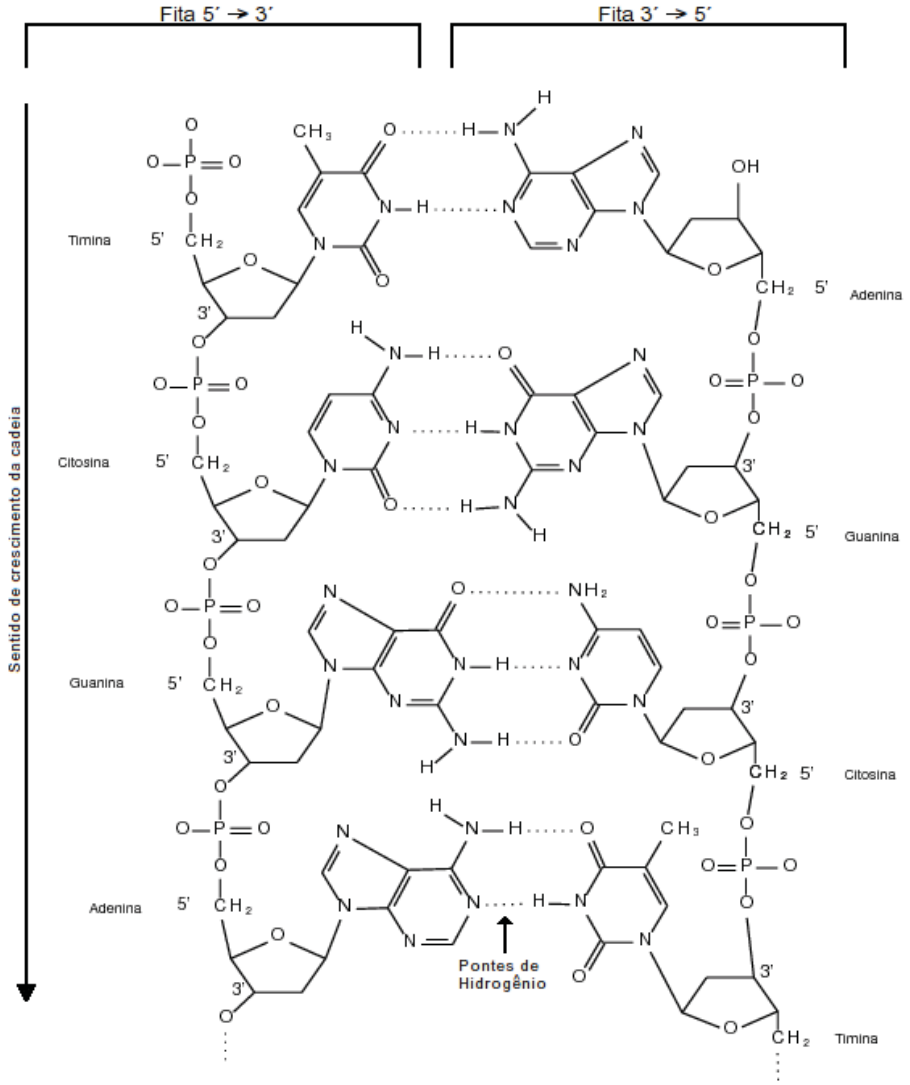

Figura 2.2: Estrutura química do DNA com as ligações fosfodiéster (entre nucleotídeos na mesma fita) e o pareamento das bases realizado por pontes de hidrogênio (pontilhado). A seta indica a orientação do crescimento da cadeia de nucleotídeos.

uma ribose e utilizar a base Uracila (U) ao invés da Timina (T) (Zaha et al. 1996).

O RNA é encontrado geralmente em fita simples (mas também ocorre em dupla hélice) e pode formar uma variedade de estruturas através do pareamento de bases entre regiões complementares da mesma fita (Higgs e Attwood 2005).

Existem vários tipos de moléculas de RNA, sendo o RNA mensageiro (mRNA), o RNA de transferência (tRNA) e o RNA ribossômico (rRNA) os três principais (Gibas e Jambeck 2001). Neste trabalho, é apresentado outro tipo de RNA, conhecido como miRNA, descoberto no início da década de 1990. O estudo dos miRNAs ganhou ênfase a partir do início dos anos 2000, quando identificou-se que estas moléculas atuam na regulação da expressão gênica dos organismos.

A tradução de RNA mensageiro em proteína é a última etapa a fim de disponibilizar as informações contidas no DNA em funcionamento na célula (Gibas e Jambeck 2001). Nesse ponto, as bases presentes no mRNA são lidas em tripletos conhecidos como códons. Cada códon corresponde a um aminoácido (Zaha et al. 1996). O conjunto de regras responsável por essa correspondência é chamado de código genético padrão (Graur e Li 2000).

Proteínas são formadas por cadeias de aminoácidos. Todas as proteínas existentes em todos os organismos vivos são construídas utilizando-se dos 20 aminoácidos apresentados na Tabela 2.1 (Graur e Li 2000). Tal construção sempre tem início com o códon `AUG` (Metionina) e é encerrada com um dos códons de terminação: `UAA`, `UGA` ou `UAG`.

Tabela 2.1: Código genético padrão

Aminoácido	**Códon do RNA**
Alanina	GCA, GCC, GCG, GCU.
Arginina	AGA, AGG, CGA, CGC, CGG, CGU.
Asparagina	AAC, AAU.
Ácido aspártico	GAC, GAU.
Cisteína	UGC, UGU.
Ácido glutâmico	GAA, GAG.
Glutamina	CAA, CAG.
Glicina	GGA, GGC, GGG, GGU.
Histidina	CAC, CAU.
Isoleucina	AUA, AUC, AUU.
Leucina	UUA, UUG, CUA, CUC, CUG, CUU.
Lisina	AAA, AAG.
Metionina	AUG.
Fenilalanina	UUC, UUU.
Prolina	CCA, CCC, CCG, CCU.
Serina	AGC, AGU, UCA, UCC, UCG, UCU.
Treonina	ACA, ACC, ACG, ACU.
Triptofano	UGG.
Tirosina	UAC, UAU.
Valina	GUA, GUC, GUG, GUU.
Códons de terminação	UAA, UAG, UGA.

O conjunto de material genético carregado por um indivíduo é chamado genoma (Graur e Li 2000), ou seja, genoma é a sequência completa do DNA presente em cada célula do organismo. Nele está contida toda a informação necessária para construir e manter vivo um exemplar deste indivíduo (Brown 1999).

O genoma é dividido em partes menores, chamadas genes (Gibas e Jambeck 2001). Tradicionalmente, gene era definido como um fragmento de DNA que codifica uma proteína.

Porém estudos mostraram a existência de três tipos de genes, e essa definição foi alterada de modo que um gene é uma sequência de DNA genômico essencial para uma função específica (Graur e Li 2000).

Os três tipos de genes são (Graur e Li 2000) (Gibas e Jambeck 2001):

- Genes que codificam proteínas: são transcritos em RNA e, em seguida, traduzidos para proteínas;
- Genes especificadores de RNA: são apenas transcritos;
- Genes não transcritos: possuem algum propósito, mas existe pouco conhecimento sobre esse tipo de gene.

Em um gene do tipo que codifica proteína, são encontrados os seguintes mecanismos:

- Códon de início e fim: marcações que indicam o início e o final da tradução, respectivamente;
- Regiões não traduzidas (*untranslated regions*): 5'UTR é a parte da sequência anterior ao códon de início e 3'UTR é a sequência após o códon de parada;
- Intron: parte do gene que não é traduzida;
- Éxon: parte do gene que é traduzida em proteína;
- Cauda Poli-A: sequência final do gene, possui apenas Adenina.

Finalmente, a Figura 2.3 apresenta o Dogma Central da Biologia Molecular. Nela estão representadas a replicação do DNA (I), a transcrição do DNA para RNA mensageiro (mRNA) (II) e a tradução do mRNA em proteína (III). A replicação e a transcrição ocorrem no núcleo e a tradução ocorre no citoplasma da célula.

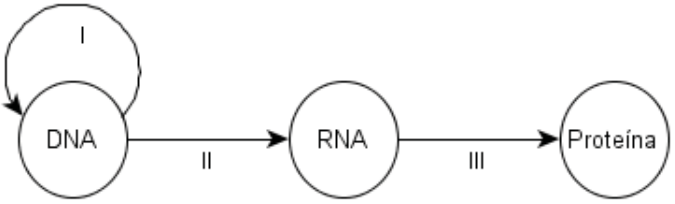

Figura 2.3: Dogma central da biologia molecular: I - replicação, II - transcrição e III - tradução

2.1.1 Biogênese do MiRNA

Segundo (Mendes, Freitas e Sagot 2009), o pre-miRNA (miRNA precursor) origina-se no núcleo da célula de dois modos principais: *i)* um miRNA primário (pri-miRNA) é processado pela enzima Drosha ou, *ii)* a partir de um tipo particular de intron, surgindo após ocorrer o processo de junção (*splicing*) em um pre-mRNA. Em seguida, o pre-miRNA é exportado ao citoplasma (pela ação de enzimas transportadoras), onde sofre o processo de maturação (enzima Dicer) no qual a volta *(loop)* existente no grampo é liberado, ficando apenas duas fitas, que formam uma estrutura conhecida como miRNA:miRNA*. Na próxima etapa, a fita miRNA é integrada ao miRISC (Complexo de Silenciamento Induzido por miRNA) e o miRNA* é degradado.

O miRISC liga-se a um RNA mensageiro alvo, influenciando o processo de tradução de dois modos: *i)* dividindo o mRNA e, consequentemente, causando sua degradação imediata ou *ii)* não permitindo que o local onde ocorreu a ligação seja traduzido, resultando em uma proteína sem função, que posteriormente será degradada. Todo o processo descrito anteriormente pode ser observado na Figura 2.4.

2.2 Evolução de Sequências

Considerando uma população, Higgs e Attwood 2005 definem evolução com base na propensão a erros e na variação do sucesso da auto-replicação dos indivíduos pertencentes a essa população. Como auto-replicação entende-se a capacidade que um organismo possui de gerar cópias de si mesmo.

A propensão a erros garante cópias nem sempre idênticas aos indivíduos originais e a variação do sucesso determina alterações na taxa de crescimento da população, pois alguns destes indivíduos terão mais chances de sobreviver e continuar a replicação (Higgs e Attwood 2005).

Erros no processo de replicação ou danos na molécula de DNA geram as mutações, que são pequenas alterações na sequência transmitidas aos descendentes (Higgs e Attwood 2005). Tais alterações podem ser realizadas por substituição, deleção ou inserção de nucleotídeos (Brown 1999).

Com base na evolução, duas sequências são ditas homólogas se ambas derivam de um ancestral comum (Duret e Abdeddaim 2000).

O alinhamento de sequências homólogas consiste em tentar colocar resíduos em colunas que derivam do resíduo de um ancestral comum. Isto é conseguido pela introdução de *gaps* (que representam inserções ou deleções) nas sequências (Duret e Abdeddaim 2000),

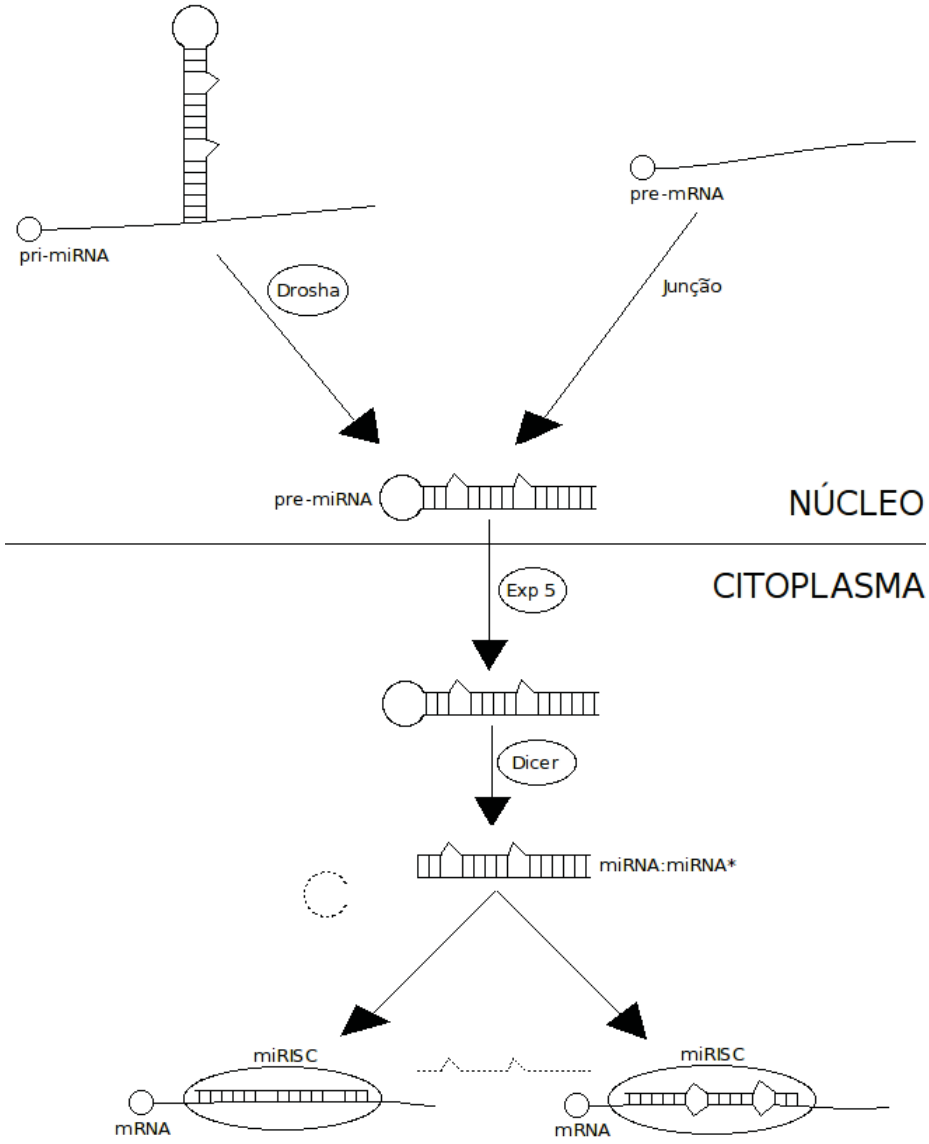

Figura 2.4: Biogênese do miRNA. A molécula pre-miRNA surge no núcleo da célula, é enviada ao citoplasma onde sofre a maturação e o miRNA liga-se ao complexo miRISC. O complexo miRISC influencia a tradução de seu RNA mensageiro alvo.

como pode ser observado pela Figura 2.5, que apresenta o alinhamento entre duas sequências hipotéticas.

```
Seq. 01: AGCCCATTTCATCCTAA
         | ||  ||||||||||||
Seq. 02: ATCC--TTTCATCCTAA
```

Figura 2.5: Alinhamento entre as sequências `AGCCCATTTCATCCTAA` e `ATCCTTTCATCCTAA`. Pareamentos são representados por "|" e *gaps* por "-".

O conceito de sequências homólogas será utilizado neste trabalho a fim de identificar miRNAs conhecidos de espécies evolucionariamente próximas no genoma da soja.

3 FUNDAMENTOS COMPUTACIONAIS

Este Capítulo tem como objetivo descrever as técnicas computacionais estudadas durante o desenvolvimento do trabalho. A Seção 3.1 descreve algoritmos utilizados em análise de similaridade e a Seção 3.2 apresenta os técnicas de inteligência artificial que realizam predição *ab initio*. A Seção 3.3 introduz conceitos de computação paralela, utilizados durante a implementação dos algoritmos.

3.1 Algoritmos para Alinhamento de Sequências

O alinhamento de sequências é uma das operações mais importantes para a bioinformática (Prosdocimi et al. 2003) e consiste em comparar duas ou mais sequências em busca de similaridades que possam existir entre elas.

Similaridade refere-se à presença de locais idênticos entre duas sequências, enquanto homologia indica que duas sequências possuem probabilidade alta de compartilharem o mesmo ancestral (Gibas e Jambeck 2001), conforme discutido na Seção 2.2.

Um exemplo clássico do alinhamento de sequências ocorre na situação onde um novo gene acaba de ser sequenciado. Nesse caso, aplica-se o alinhamento dessa nova sequência contra um banco de dados de proteínas a fim de estabelecer sua função no organismo.

O alinhamento entre sequencias pode ser realizada de dois modos: *i)* Global – o qual considera-se o tamanho total das sequências, produzindo apenas um resultado ou; *ii)* Local – onde pequenas subsequências são comparadas e, consequentemente, cada fragmento apresenta um resultado.

3.1.1 Basic Local Alignment Search Tool – BLAST

A tarefa de encontrar similaridade entre sequências é complexa e, geralmente, é realizada por programas especializados. O algoritmo mais difundido nessa área é o BLAST, do inglês *Basic Local Alignment Search Tool*, que foi desenvolvido por Altschul et al. 1990 e utiliza-se do alinhamento local. O BLAST pode executar milhares de comparações entre

sequências em minutos (Gibas e Jambeck 2001).

Este algoritmo mantém uma matriz de pontuação de similaridade entre todos os pares de resíduos, a qual é preenchida com valores positivos para regiões conservadas entre as sequências comparadas e com valores negativos nas regiões não conservadas (Altschul et al. 1990).

Segundo Gibas e Jambeck 2001, o alinhamento com o algoritmo BLAST possui três etapas: *i)* cria-se uma lista com todas as palavras curtas com pontuação acima de um valor limite, após alinhadas com a sequência de pesquisa; *ii)* Consulta-se o banco de dados para obter as ocorrências destas palavras, que são combinadas e estendidas para alinhamentos locais entre a sequência de pesquisa e a sequência do banco da dados; *iii)* Os alinhamentos com pontuação mais alta são combinados em alinhamentos locais, onde possível.

A Figura 3.1 apresenta o resultado do alinhamento da sequência exemplo `ATGGCAGGAGCTGCACCACCACCCAAGCCAGAAGAGCTTCAGCCA`, contra o banco de dados de pre-miRNA de plantas, realizado através do endereço `http://bioinformatics.cau.edu.cn/PMRD/`. Nota-se que ocorreu alinhamento local de 13 bases com um fragmento da sequência identificada por 'tae-MIR1128'.

```
>tae-MIR1128
          Length = 188
 Score = 26.3 bits (13), Expect = 0.42
 Identities = 13/13 (100%)
 Strand = Plus / Minus

Query: 23  ccaagccagaaga 35
           |||||||||||||
Sbjct: 179 ccaagccagaaga 167
```

Figura 3.1: Exemplo de saída do programa BLAST, no qual uma sequência exemplo foi alinhada contra pre-miRNAs de plantas.

3.1.2 Short Oligonucleotide Alignment Program – SOAP

O *Short Oligonucleotide Alignment Program (SOAP)*, desenvolvido por (Li et al. 2008), é outro algoritmo para alinhamento de sequências, projetado para trabalhar com sequências curtas, que é uma característica de algumas das novas tecnologias de sequenciamento, como a Solid^{TM} e a Solexa. A versão atual do pacote, disponível para *download* no endereço `http://soap.genomics.org.cn/`, possui, além de outras ferramentas, o programa SOAPaligner/SOAP2, o qual implementa o algoritmo SOAP.

Durante o alinhamento, o algoritmo SOAP executa as seguintes etapas: *i)* carrega as sequências de referência para a memória RAM; *ii)* cria tabelas *hash* para a indexação das sequências semente; *iii)* Para cada sequência de referência, faz a busca pelos acertos nas sequências semeadas, realizando, desse modo, o alinhamento.

O SOAPaligner/SOAP2 existe apenas para arquiteturas de 64 bits devido à necessidade de grande quantidade de memória RAM no momento da criação de suas tabelas de indexação.

3.2 Técnicas de Aprendizagem de Máquina

A predição de pre-miRNAs ainda não descobertos, também conhecida como predição *ab initio*, pode ser realizada com o uso de técnicas de aprendizagem de máquina. Durante o desenvolvimento deste trabalho, foram estudados os algoritmos Máquina de Vetor de Suporte, do inglês *Support Vector Machine (SVM)* (Seção 3.2.1) e as Cadeias de Markov (Seção 3.2.2). O primeiro foi implementado e utilizado para a predição de novos pre-miRNAs na soja, enquanto que o segundo será utilizado em trabalhos futuros.

3.2.1 Máquina de Vetor de Suporte

Máquina de Vetor de Suporte, do inglês *Support Vector Machine (SVM)*, é um algoritmo de aprendizagem de máquina embasado pela teoria do aprendizado estatístico, a qual estabelece uma série de princípios a serem seguidos na obtenção de classificadores com boa generalização (Lorena e Carvalho 2007).

A SVM apresenta o estado da arte em performance de classificação para uma grande variedade de domínios de aplicação (Noble 2004) e, recentemente, vem sendo aplicado com sucesso no campo da bioinformática.

Na SVM, o aprendizado ocorre de maneira supervisionada, de modo que, durante a fase de treinamento ou aprendizagem, padrões com respostas conhecidas são apresentados ao algoritmo. Ao final do treinamento, espera-se que a SVM seja capaz de predizer a que classe pertence uma nova entrada de dado, que ainda não lhe foi apresentada. Conforme (Lorena e Carvalho 2007), o classificador obtido também pode ser visto como uma função f, a qual recebe um dado x e fornece uma predição y.

3.2.1.1 Hiperplano Ótimo para Padrões de Dados Linearmente Separáveis

Considerando um problema onde existem duas classes de dados linearmente separáveis, no qual os dados são representados por $\Im = (x_i, d_i)$ (padrão de entrada, valor

esperado), a equação da superfície de decisão na forma de um hiperplano que realiza tal separação é (Haykin 1999):

$$w^T x + b = 0 \tag{3.1}$$

onde x é o vetor de entrada, w o vetor de pesos e b um bias.

O objetivo de uma Máquina de Vetor de Suporte é encontrar o hiperplano particular para o qual a margem de separação ρ é máxima (Haykin 1999). Margem de separação entende-se como a distância entre o hiperplano e o ponto de dado mais próximo (Platt 1998).

A Figura 3.2, adaptada de (Haykin 1999) apresenta um hiperplano ótimo para padrões de dados pertencentes a duas classes linearmente separáveis.

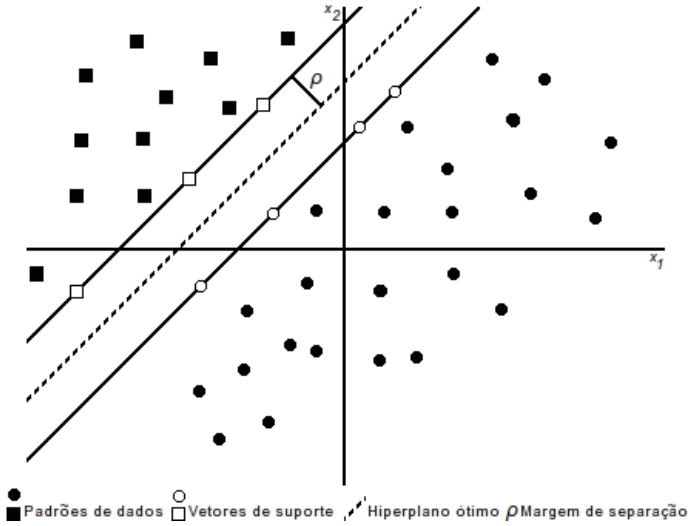

Figura 3.2: Hiperplano ótimo para duas classes de dados linearmente separáveis.

A fim de estabelecer o hiperplano ótimo, deve-se encontrar os parâmetros w_0 e b_0, a partir do conjunto de treinamento, que satisfaçam a restrição (Burges 1998) (Haykin 1999):

$$\begin{aligned} {w_0}^T x_i + b_0 &\geq 1, \forall d_i = +1 \\ {w_0}^T x_i + b_0 &\leq -1, \forall d_i = -1 \end{aligned} \tag{3.2}$$

Os pontos de dados que estão sobre as linhas de separação na Figura 3.2 satisfazem a primeira ou a segunda linha da Equação 3.2 pela igualdade são denominados vetores de suporte e influenciam diretamente a localização ótima da superfície de decisão (Burges 1998) (Haykin 1999).

A busca do hiperplano ótimo torna-se um problema de otimização restrito, conhecido como problema primordial que é definido como segue (Haykin 1999): dada a amos-

tra de treinamento $(x_i, d_i)_{i=1}^N$, encontrar os valores ótimos para o vetor peso w e o bias b de modo que as restrições presentes na Equação 3.3 sejam satisfeitas:

$$d_i(w^T x_i + b) \geq 1, i = 1, 2, ..., N \tag{3.3}$$

e que a função de custo mostrada na Equação 3.4 seja minimizada pelo vetor peso w:

$$\Phi(w) = \frac{1}{2} w^T w \tag{3.4}$$

onde a função de custo $\Phi(w)$ é convexa e as restrições são lineares a w.

A partir de um problema de otimização restrito como o primordial, pode-se construir um outro problema, chamado problema dual, de acordo com a formulação a seguir (Haykin 1999): dada a amostra de treinamento $(x_i, d_i)_{i=1}^N$, encontrar os multiplicadores de Lagrange $\{\alpha_i\}_{i=1}^N$, que maximizem a função objetivo mostrada na Equação 3.5:

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j {x_i}^T x_j \tag{3.5}$$

sujeito às restrições apresentadas nas Equações 3.6 e 3.7:

$$\sum_{i=1}^{N} \alpha_i d_i = 0 \tag{3.6}$$

$$\alpha_i \geq 0, i = 1, 2, ..., N \tag{3.7}$$

De acordo com Burges 1998, os multiplicadores de Lagrange são utilizados por duas razões: *i)* facilidade de manuseio e; *ii)* os dados de treinamento aparecem somente na forma de produto escalar entre vetores, permitindo a generalização para o caso não linear.

3.2.1.2 Hiperplano Ótimo para Padrões de Dados não Linearmente Separáveis

A aplicação do algoritmo desenvolvido para dados linearmente separáveis não produz bons resultados quando aplicado em conjuntos de dados não linearmente separáveis (Burges 1998), ou seja, não é possível construir um hiperplano de separação sem que ocorram erros de classificação. Logo, buscam-se modificações que permitam o trabalho com dados não separáveis.

A margem de classificação é considerada suave se um ponto de dado (x_i, d_i) violar a condição representada pela Equação 3.8 (Haykin 1999) e pode ocorrer quando o ponto de dado encontra-se dentro da região de separação, mas do lado correto da superfície

de decisão ou quando o ponto de dado está do lado errado da superfície de decisão.

$$d_i(w^T x_i + b) \geq +1, i = 1, 2, ..., N \tag{3.8}$$

Para o caso de pontos de dados não separáveis, a Equação 3.8 pode ser reescrita com o uso de variáveis soltas positivas (Burges 1998), que medem o desvio de um ponto de dado da condição ideal de separabilidade de padrões, como pode ser observado na Equação 3.9 (Burges 1998) (Haykin 1999):

$$d_i(w^T x_i + b) \geq +1 - \xi_i, i = 1, 2, ..., N \tag{3.9}$$

A função custo a ser minimizada em relação ao vetor peso w é mostrada na Equação 3.10 (Haykin 1999):

$$\Phi(w, \xi) = \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i \tag{3.10}$$

onde o parâmetro C, ou parâmetro de regularização, controla o compromisso entre a complexidade da máquina e o número de pontos não separáveis, determinado de modo experimental ou analítico (Haykin 1999) no qual um alto valor para esse parâmetro estabelece grande penalidade para erros de classificação (Burges 1998).

O problema primordial para casos não separáveis pode ser definido como segue (Haykin 1999): dada uma amostra de treinamento $\{(x_i, d_i)\}_{i=1}^{N}$ encontrar valores ótimos para o vetor peso w e para o bias b que satisfaçam à restrição definida na Equação 3.9, onde $\xi \geq 0, \forall i$ e o vetor peso w juntamente com as variáveis soltas ξ_i minimizem a função custo da Equação 3.10.

O problema dual, para casos de padrões de dados não separáveis pode ser formulado como segue (Burges 1998) (Haykin 1999): dada uma amostra de treinamento, $\{(x_i, d_i)\}_{i=1}^{N}$, encontrar os multiplicadores de Lagrange $\{\alpha_i\}_{i=1}^{N}$ que maximizam a função objetivo, apresentada pela Equação 3.11:

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j x_i^T x_j \tag{3.11}$$

sujeita às restrições descritas na Equações 3.12 e 3.13:

$$\sum_{i=1}^{N} \alpha_i d_i = 0 \tag{3.12}$$

$$0 \leq \alpha_i \leq C, i = 1, 2, ..., N \tag{3.13}$$

3.2.1.3 Núcleo do Produto Interno

A construção da Máquina de Vetor de Suporte, além de determinar o hiperplano ótimo para a separação dos padrões, deve passar por uma etapa anterior, na qual ocorre o mapeamento não-linear do vetor de entrada para um espaço de maior dimensão (Lorena e Carvalho 2007).

Seja $\varphi : X \rightarrow \Im$ um mapeamento onde X é o espaço de entradas e $\Im$ o espaço de características. A escolha apropriada de φ faz com que o conjunto de treinamento mapeado em $\Im$ possa ser separado por uma Máquina de Vetor de Suporte linear (Lorena e Carvalho 2007). A Figura 3.3, retirada de (Haykin 1999) ilustra o mapeamento dos dados a partir do espaço de entrada para o espaço de características, realizado por φ.

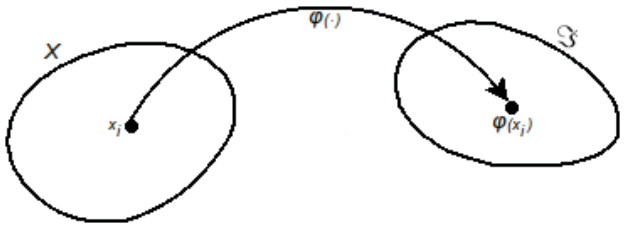

Figura 3.3: Mapeamento do espaço de entrada para um espaço de maior dimensão, através da função $\varphi(\cdot)$.

A Equação 3.14, define a superfície de decisão calculada no espaço de características em termos de pesos lineares da Máquina de Vetor de Suporte (Haykin 1999):

$$\sum_{j=0}^{m_i} w_j \varphi_j(x) = 0 \tag{3.14}$$

onde m_0 denota a dimensão do espaço de entrada, m_1 é a dimensão do espaço de características e o vetor $\varphi(x)$ representa a imagem induzida no espaço de características pelo vetor de entrada x.

O núcleo do produto interno representado por $K(x, x_i)$, também conhecido como função *Kernel*, é definido pela Equação 3.15 (Haykin 1999):

$$K(x, x_i) = \sum_{j=0}^{m_1} \varphi_j(x)\varphi_j(x_i), i = 1, 2, ..., N \tag{3.15}$$

O núcleo do produto interno pode ser utilizado para construir o hiperplano ótimo no espaço de características sem ter que considerar o próprio espaço de características de forma explícita, ou seja, o mapeamento é gerado implicitamente (Lorena e Carvalho 2007), como pode ser observado na Equação 3.16, que é um caso especial do teorema de Mercer

(Haykin 1999):

$$\sum_{i=i}^{N} \alpha_i d_i K(x, x_i) = 0 \tag{3.16}$$

Segundo Lorena e Carvalho 2007, para a construção da SVM, utiliza-se um núcleo que segue as condições do Teorema de Mercer, para garantir: *i)* a convexidade do problema de otimização presente na Equação 3.11 e; *ii)* que os mapeamentos nos quais ocorram o cálculo de produtos escalares sejam realizados conforme e Equação 3.15.

Talvez a maior limitação da abordagem de Máquina de Vetor de Suporte seja a escolha do núcleo de produto interno a ser utilizado (Burges 1998). A Tabela 3.1 apresenta as funções *Kernel* frequentemente mais encontrados na literatura.

Tabela 3.1: Núcleos de produto interno encontrados com maior frequência na literatura.

Tipo de Máquina de Vetor de Suporte	**Núcleo de produto interno**
Máquina de aprendizagem polinomial	$(x^T x_i + 1)^p$
Rede de função de base radial	$exp(-\frac{1}{2\sigma^2}\|x - x_i\|^2)$
Rede neural sigmoidal de duas camadas	$tanh(\beta_0 x^T x_i + \beta_1)$

Conforme Haykin 1999, a formulação do problema dual para a otimização restrita de uma Máquina de Vetor de Suporte pode ser realizada como segue: dada uma amostra de treinamento $\{(x_i, d_i)\}$, encontrar os multiplicadores de Lagrange $\{\alpha_i\}_{i=1}^N$ que maximizam a função objetivo mostrada na Equação 3.17:

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j K(x_i, x_j) \tag{3.17}$$

Sujeito às restrições apresentadas nas Equações 3.18 e 3.19:

$$\sum_{i=1}^{N} \alpha_i d_i = 0 \tag{3.18}$$

$$0 \leq \alpha_i \leq C, i = 1, 2, ..., N \tag{3.19}$$

Finalmente, a Figura 3.4, retirada de (Haykin 1999), apresenta a arquitetura da Máquina de Vetor de Suporte.

3.2.1.4 Algoritmos para a resolução do problema dual

Os algoritmos que resolvem o problema dual também são conhecidos como algoritmos de treinamento para a SVM. Segundo Ales 2008 técnicas como o Método do Gradiente, o Método de Newton e o Método de Quase-Newton convergem para uma solução após

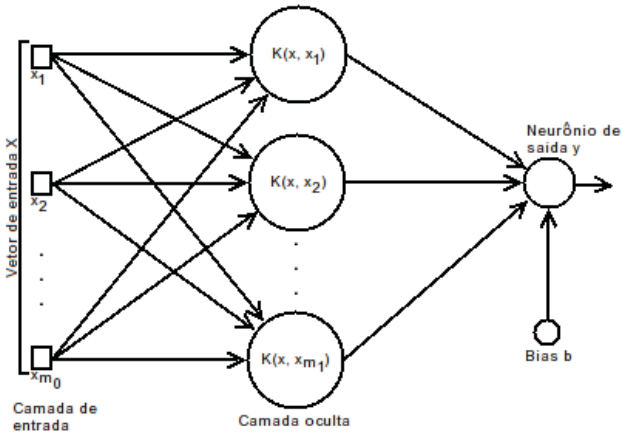

Figura 3.4: Arquitetura para a Máquina de Vetor de Suporte.

um certo número de passos, porém, são caros do ponto de vista computacional, fato que pode torná-los inviáveis para problemas com grande quantidade de dados, além de possuírem alta complexidade de implementação.

Por esse motivo, segundo Lorena e Carvalho 2007, na solução do problema dual para aplicações de larga escala, a estratégia de decomposição geralmente é utilizada.

Nesse contexto, o algoritmo *Sequential Minimal Optimization (SMO)* Platt 1998, que é um caso especial do algoritmo de Osuna (Osuna, Freund e Girosi 1997), apresenta-se como um algoritmo para o treinamento da SVM de fácil implementação além de ser baseado em heurísticas que aceleram sua execução.

Conforme Platt 1998, o algoritmo SMO decompõe o problema quadrático inicial em sub-problemas quadráticos menores, onde, a cada iteração, são escolhidos apenas dois multiplicadores de Lagrange para otimizar. Logo, o algoritmo SMO é dividido em dois componentes: *i)* uma heurística utilizada para escolher quais multiplicadores otimizar na iteração atual e; *ii)* um método analítico para resolver o problema para os multiplicadores escolhidos.

3.2.2 Cadeias de Markov

Esta Seção introduz um caso particular de Processo Estocástico, as Cadeias de Markov, uma segunda técnica de aprendizagem de máquina estudada durante o desenvolvimento desse trabalho.

As Cadeias de Markov são chamadas assim por apresentarem a propriedade Markoviana (em homenagem ao matemático russo Andrei Andreyevich Markov). Inicialmente, seguem as definições de Variável Aleatória e Processo Estocástico:

- **Variável Aleatória:** é uma função X que, em um experimento E de um espaço amostral

S, associa cada elemento $s \in S$ a um valor real $x = X(s)$.

- **Processo Estocástico:** coleção de variáveis aleatórias indexadas por um parâmetro $\{X(t), t \in T\}$. Logo, T é o conjunto de índices do processo em questão. Se, por exemplo, t representar o tempo, $X(t)$ define o estado do processo no tempo t.

Seja um sistema em que sua evolução é descrita por um processo estocástico, por exemplo, uma sequência de experimentos cujos possíveis resultados são $E_1, E_2, ..., E_n$. Se as probabilidades dessa sequência de resultados são definidas por:

$$P(E_{j_0}, E_{j_1}, ..., E_{j_n}) = a_{j_0} p_{j_0 j_1} p_{j_1 j_2} ... p_{j_{n-2} j_{n-1}} p_{j_{n-1} j_n} \tag{3.20}$$

em termos de uma distribuição de probabilidade $\{a_k\}$ para E_k no experimento inicial e fixando probabilidades condicionais p_{jk} de E_k dado que E_j ocorreu no experimento anterior, então essa sequência de experimentos é chamada processo de Markov (Feller 1968).

Um processo de Markov discreto no tempo com espaço de estados discreto é denominado **Cadeia de Markov** (Cox e Miller 1977).

Segundo (Feller 1968), os possíveis resultados E_k representam os possíveis estados que o sistema pode alcançar e p_{jk} denota a probabilidade de ocorrer a transição do estado E_j para o estado E_k. Considerando a_k como a probabilidade do estado E_k ser escolhido como inicial, logo $a_k \geq 0$ e $\sum a_k = 1$.

A Equação (3.20) é conhecida como propriedade Markoviana e indica que uma sequência de variáveis aleatórias forma uma cadeia de Markov se a probabilidade que o sistema tem de chegar ao estado E_k no instante $n+1$ depende exclusivamente da probabilidade do sistema estar no estado E_j no instante n (Haykin 1999).

As probabilidades de transição p_{jk} são probabilidades condicionais e devem satisfazer as seguintes Equações (Haykin 1999):

$$p_{jk} \geq 0, \forall j, k \tag{3.21}$$

$$\sum_k p_{jk} = 1, \forall j \tag{3.22}$$

Tais probabilidades podem ser expressas por meio de uma matriz quadrada ($K \times K$, na qual K denota o número de estados), chamada matriz de probabilidades de

transição (Feller 1968):

$$P = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1K} \\ p_{21} & p_{22} & \cdots & p_{2K} \\ \vdots & \vdots & & \vdots \\ p_{K1} & p_{K2} & \cdots & p_{KK} \end{bmatrix}$$

As Equações (3.21) e (3.22) determinam que essa matriz é formada por valores não negativos e que a soma de cada linha é igual a 1. Uma matriz desse tipo é chamada Matriz Estocástica (Haykin 1999).

A matriz de probabilidades, juntamente com a distribuição inicial $\{a_k\}$ define completamente uma cadeia de Markov com estados $E_1, E_2, ...$ (Feller 1968).

Exemplo: Cadeia de Markov com dois estados

O exemplo a seguir foi retirado de (Cox e Miller 1977) e apresenta a matriz de probabilidade de transição para uma Cadeia de Markov com dois estados: sucesso, denotado por 1 ou fracasso, indicado por 0. A probabilidade de sucesso ou fracasso do experimento atual depende do resultado do experimento anterior. Supondo que se o $n-$ésimo experimento resultou em fracasso, então as probabilidades de fracasso e sucesso do $(n+1)-$ ésimo experimento são $1-\alpha$ e α, respectivamente. De modo similar, se o $n-$ésimo experimento resultou em sucesso, então as probabilidades de sucesso e fracasso do $(n+1)-$ésimo experimento são, respectivamente, $1-\beta$ e β. A matriz das probabilidades de transição é apresentada a seguir:

$$P = \begin{bmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{bmatrix}$$

A Figura 3.5 exibe o diagrama de transição de estados para a Cadeia de Markov do exemplo anterior.

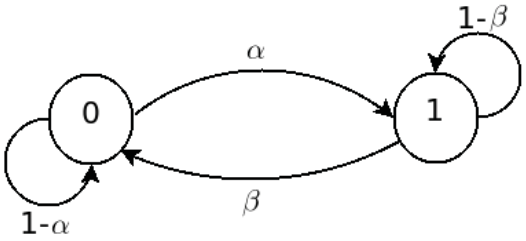

Figura 3.5: Diagrama de transição de estados.

A probabilidade de transição no caso em que são necessários exatos n passos para, saindo do estado E_j chegar ao estado E_k, é denotada por $p_{jk}^{(n)}$. Assim, de acordo com

(Feller 1968):

$$p_{kj}^{(1)} = p_{jk} \tag{3.23}$$

$$p_{jk}^{(2)} = \sum_{v} p_{jv} p_{vk} \tag{3.24}$$

Por indução:

$$p_{jk}^{(n+1)} = \sum_{v} p_{jv} p_{vk}^{(n)} \tag{3.25}$$

Logo, dadas as probabilidades iniciais $p^{(1)}$ e a matriz de transição de probabilidades P, pode-se utilizar a Equação 3.25 para obter as probabilidades de transição entre os estados a qualquer instante n (Cox e Miller 1977).

Generalizando a Equação 3.25, obtém-se:

$$p_{jk}^{(m+n)} = \sum_{v} p_{jv}^{(m)} p_{vk}^{(n)} \tag{3.26}$$

que é um caso especial da identidade Chapman-Kolmogorov (Feller 1968). A Equação 3.26 indica que, para uma cadeia de Markov inicialmente no estado E_j, visitar o estado E_k após $(n+m)$ passos, é necessário que esta visite um estado intermediário E_v após m passos e, n passos depois, visite o estado E_k.

Os elementos $p_{jk}^{(n)}$ podem ser arranjados em uma matriz, denotada por P^n. Simbolicamente, $P^{n+1} = PP^n$ e $P^{(n+m)} = P^m P^n$ (Feller 1968).

3.2.2.1 Classificação dos Estados

Chama-se estado de período d o estado em uma Cadeia de Markov que ocorre em intervalos de tempo múltiplos de um valor inteiro d. Se todos os estados da cadeia possuírem período igual a 1, esta é aperiódica (Haykin 1999).

Um estado recorrente (persistente) é aquele que, uma vez alcançado, possui probabilidade igual a 1 de ocorrer novamente. Neste caso, o tempo de recorrência será uma variável aleatória e o estado será chamado recorrente positivo se o tempo for finito ou recorrente nulo, quando o tempo for infinito. Se a probabilidade for menor que 1, o estado é chamado transiente (Cox e Miller 1977). Se uma Cadeia de Markov possuir alguns estados transientes e alguns estados recorrentes, então, eventualmente, o processo irá mover-se apenas entre os estados recorrentes (Haykin 1999). Um estado aperiódico recorrente positivo é chamado ergódico (Cox e Miller 1977).

Um estado E_k é dito efêmero se $p_{jk} = 0$ para qualquer j. Logo, uma cadeia pode estar em um estado efêmero apenas se este for seu estado inicial (Cox e Miller 1977).

O estado E_k de uma Cadeia de Markov é dito ser acessível (alcançável) de um estado E_j se existir uma sequência finita de transições de E_j para E_k com probabilidade positiva. Se os estados E_k e E_j são acessíveis entre si, então também são comunicantes (Haykin 1999).

Um conjunto de estados C é fechado se nenhum estado fora de C, em uma Cadeia de Markov, pode ser alcançado partindo de qualquer estado E_j em C (Feller 1968). Se o conjunto C possuir um único estado E_k, tal estado será chamado absorvente. Um conjunto C_m de estados é dito Conjunto Fechado Mínimo se este não possuir subconjuntos fechados.

Dois estados pertencem à mesma classe se ambos possuem o mesmo período ou ambos são aperiódicos; se ambos são transientes ou senão persistentes; ou se ambos possuem tempo finito de recorrência ou então possuem tempo infinito de recorrência (Feller 1968). Em geral, as Cadeias de Markov consistem de uma ou mais classes disjuntas (Haykin 1999) e podem ser classificadas em:

- Cadeia Irredutível: quando todos os estados da cadeia pertencem à mesma classe (Haykin 1999) (Feller 1968);
- Cadeia Redutível: o oposto da irredutível, ou seja, existem pelo menos duas classes de estados distintas na cadeia.
- Ergódica: quando todos os estados da cadeia são ergódicos. No contexto de Cadeias de Markov, ergodicidade significa que a proporção de tempo gasto pela cadeia no estado E_j corresponde à probabilidade de estado-estável do estado E_j, denotada por a_{E_j} (Haykin 1999). A proporção de tempo gasto no estado E_j após k retornos, denotado por $v_{E_j}(k)$ é definida por:

$$v_{E_j}(k) = \frac{k}{\sum_{l=1}^{k} T_{E_j}(l)} \tag{3.27}$$

  onde $T_{E_j}(l)$ denota o tempo gasto entre os retornos anterior e atual ao estado E_j. Cada retorno é estatisticamente independente e, quando E_j é um estado recorrente, a cadeia retorna a ele um número infinito de vezes. Logo, quando o número de retornos aproxima-se do infinito, de acordo com a Lei dos Grandes Números, a proporção de tempo gasto no estado E_j aproxima-se da probabilidade de estado-estável, como mostrado a seguir:

$$\lim_{k\to\infty} v_{E_j}(k) = a_i, \forall i = 1, 2, ..., K \tag{3.28}$$

  Uma condição suficiente, porém não necessária para uma Cadeia de Markov ser considerada ergódica é ela ser irredutível e aperiódica (Haykin 1999).

- Absorvente: Uma Cadeia de Markov é dita absorvente se possui pelo menos um estado absorvente que pode ser alcançado (em n passos) por qualquer outro estado.

3.3 Computação Paralela

O modelo de Von Neumann divide o computador digital em cinco partes principais: *i)* Unidade de entrada; *ii)* Unidade de memória; *iii)* Unidade aritmética e lógica; *iv)* Unidade de controle e; *v)* Unidade de saída.

A unidade central de processamento *(Central Processing Unit - CPU)*, também conhecida como processador, é composta pelo agrupamento da unidade de controle com a unidade aritmética e lógica. É responsabilidade da CPU a execução e o controle das instruções de programas carregados na memória.

Conforme a Taxonomia de Flynn, os sistemas de computação são classificados em quatro grupos, de acordo com os fluxos de instruções e dados: *i)* Único Fluxo de Instruções, Único Fluxo de Dados (*Single Instruction Stream, Single Data Stream – SISD*) – máquinas mono processadas; *ii)* Único Fluxo de Instruções, Múltiplos Fluxos de Dados (*Single Instruction Stream, Multiple Data Stream – SIMD*) – máquinas vetoriais e matriciais; *iii)* Múltiplos Fluxos de Instruções, Único Fluxo de Dados (*Multiple Instruction Stream, Single Data Stream – MISD*) – não possui implementação e; *iv)* Múltiplos Fluxos de Instruções, Múltiplos Fluxos de Dados (*Multiple Instruction Stream, Multiple Data Stream – MIMD*) – multiprocessadores simétricos e *clusters*. As arquiteturas paralelas, de acordo com a Taxonomia de Flynn, são representadas pelos grupos *ii* e *iv*.

Logo, de acordo com Cáceres, Mongelli e Song 2001, um sistema de computação paralela consiste de uma coleção de elementos de computação (CPUs), interconectados conforme uma determinada topologia a fim de permitir a coordenação de suas atividades e troca de dados. É senso comum que a evolução dos computadores, passará por um caminho onde cada vez mais é explorado o paralelismo (Pasin e Kreutz 2003).

A classe das máquinas MIMD ainda pode ser subdividida em outras duas, conforme o tipo de acesso à memória RAM (Rose e Navaux 2003). São elas os multiprocessadores, máquinas onde duas ou mais CPUs compartilham uma única memória e os multicomputadores, máquinas nas quais cada CPU possui sua própria memória.

Nos multiprocessadores, todos os processadores P acessam, através de uma rede de interconexão, uma memória compartilhada M (Rose e Navaux 2002). Existe um único espaço de endereçamento, sendo possível a cada processador P endereçar qualquer memória M. Logo, operações do tipo *load* e *store* são suficientes para realizar a comunicação entre os processos.

Essa arquitetura recebe o nome de multiprocessador pois, comparando com uma arquitetura convencional (SISD), apenas o componente processador é replicado (Rose e Navaux 2002)

Já os multicomputadores são máquinas paralelas onde toda a arquitetura con-

vencional é replicada. Desse modo, cada posição de cada memória local *M* possui uma identificação (endereço) e pode ser acessada apenas por seu processador *P* associado (Cáceres, Mongelli e Song 20

O desenvolvimento de algoritmos para execução nessa arquitetura, necessita utilizar construções adicionais do tipo *send* e *receive* as quais possibilitam a comunicação entre os processadores. Tal característica faz esse tipo de máquina também ser conhecida como sistemas de troca de mensagens (*message passing systems*) (Rose e Navaux 2002).

Para que o paralelismo ocorra, além da existência de mais de uma CPU, é necessário que o sistema operacional seja preparado para gerenciar o ambiente paralelo e, que os programas sejam escritos de modo a reconhecer e utilizar tal ambiente, seja ele uma arquitetura multiprocessador ou multicomputador.

A execução em paralelo de um algoritmo permite, dependendo das características do problema a ser resolvido por ele, ganhos significativos de performance. Em geral, aplicações de bioinformática necessitam processar grandes quantidades de dados e, por isso, as técnicas de computação paralela são muito utilizadas nessa área.

No caso da identificação de pre-miRNAs através de técnicas de aprendizagem de máquina, a computação paralela foi utilizada durante o pré-processamento dos dados e também na fase de treinamento do algoritmo SVM. No caso do pré-processamento, o tempo médio para calcular os atributos de cada sequência foi de sete minutos. Logo, o uso de computação paralela mostrou-se determinante para a diminuição do tempo de execução nesta etapa. Os detalhes do ambiente de execução e da implementação dos *scripts* estão descritos no Capítulo 4 desta dissertação.

4 METODOLOGIA

Neste capítulo será descrita a metodologia utilizada para a descoberta de sequências de pre-miRNA no genoma da soja (*Glycine max*). As abordagens utilizadas aqui foram a busca por homologia, apresentada na Seção 4.1 e o uso de aprendizagem de máquina, descrito na Seção 4.2.

A busca por homologia foi utilizada a fim de identificar, na soja, sequências de pre-miRNA conhecidas de outras plantas, com o uso de algoritmos de alinhamento de sequências. Já o uso de técnicas de aprendizagem de máquina objetivou a descoberta de novas sequências de pre-miRNAs no genoma da soja.

4.1 Busca por Homologia

Nesta Seção é apresentada a metodologia utilizada a fim de descobrir, no genoma da soja, a ocorrência de miRNAs já conhecidos em espécies de plantas evolucionariamente próximas. A Seção 4.1.1 apresenta o conjunto de dados utilizado para a comparação e a Seção 4.1.2 mostra os passos realizados durante o alinhamento das sequências na busca por homologia.

4.1.1 Conjuntos de Dados para a Busca por Homologia

O conjunto de dados utilizado para a busca por homologia é formado por todos os pre-miRNAs conhecidos de plantas, totalizando 8998 sequências, pertencentes 120 a organismos e foi obtido no site Plant MiRNA Database[1] (Zhang et al. 2010).

O genoma da soja, o qual serviu de referência para a busca por homologia, foi sequenciado por Schmutz et al. 2010 e obtido no site *Phytozome*[2], em sua versão 7.0.

[1] `http://bioinformatics.cau.edu.cn/PMRD/`
[2] `www.phytozome.net`

4.1.2 Alinhamento Usando o Programa SOAPaligner/SOAP2

Como as sequências de pre-miRNA utilizadas aqui são relativamente curtas (possuem média de 125 bases de comprimento), o alinhamento foi realizado com o programa *Short Oligonucleotide Analysis Package (SOAP2*[3]*)* (Li et al. 2008) na versão 2.21.

A sequência pre-miRNA de outro organismo foi considerada presente na soja caso o alinhamento, com o uso do SOAP2aligner/SOAP2, tenha, no máximo, quatro diferenças de base.

4.2 Uso de Aprendizagem de Máquina

A metodologia empregada para a identificação de novos pre-miRNAs no genoma da soja, com o uso do algoritmo SVM, será apresentada nesta seção.

Como o classificador desenvolvido é utilizado para investigar a existência de pre-miRNAs no genoma da soja, os dados utilizados (Seção 4.2.1) foram obtidos de pre-miRNAs conhecidos de plantas (conjunto positivo) e outros tipos de ncRNA e sequências aleatórias (conjunto negativo) (Seção 4.2.1). Os atributos das sequências (Seção 4.2.2) foram determinados como no trabalho de Loong e Mishra 2007, pois este apresentou bons resultados quanto à exatidão, sensibilidade e especificidade. A Seção 4.2.3 descreve os detalhes de implementação e o ambiente de execução dos algoritmos. Os dados foram submetidos a uma etapa de pré-processamento (Seção 4.2.4) e, em seguida, utilizados para o treinamento do algoritmo de aprendizagem de máquina e posterior classificação das sequências candidatas a pre-miRNA (Seção 4.2.5). As sequências classificadas como pre-miRNA foram verificadas contra sequências que codificam proteína, pertencentes ao genoma da soja (Seção 4.2.6).

4.2.1 Conjuntos de Dados para a Busca com Aprendizagem de Máquina

O conjunto positivo foi criado com um total de 430 sequências pre-miRNAs conhecidas de plantas, obtidos de Plant MicroRNA Database[4] (Zhang et al. 2010), das quais 131 são pre-miRNAs de soja (*Glycine max*), 199 são de *Arabidopsis thaliana* e 100 são de *Medicago truncatula*.

O conjunto negativo foi composto por 400 sequências, das quais 175 são pequenos RNAs nucleares (*small nuclear RNA - snoRNA*) pertencentes à *Arabidopsis thaliana*,

[3] `http://soap.genomics.org.cn/`
[4] `http://bioinformatics.cau.edu.cn/PMRD/`

obtidos da deepBase [5] (Yang et al. 2010) e outras 225 sequências de RNA geradas aleatoriamente.

As sequências candidatas a pre-miRNA foram determinadas de dois modos: i) através de uma varredura completa no genoma da soja na versão Glyma1[6](Schmutz et al. 2010), em busca de sequências que podem apresentar estrutura secundária em formato de grampo (*hairpin*), como mostrado na Figura 4.1 e ii) a partir de introns existentes em transcritos conhecidos da soja (*Glycine max*), obtidos com o uso da ferramenta para estudos comparativos de genoma Phytozome[7] através da opção Biomart[8] e pré-processadas, conforme descrito na Seção 4.2.4.

Figura 4.1: Exemplo de estrutura secundária do miRNA (pre-miRNA) em formato de grampo.

4.2.2 Conjunto de Atributos das Sequências

Os conjuntos de dados descritos anteriormente possuem apenas sequências de nucleotídeos, ou seja, são formados apenas por *strings* de caracteres. Porém, o algoritmo SVM necessita de uma entrada numérica. Logo, é necessário codificar as sequências a fim de possibilitar o trabalho do algoritmo SVM. Desse modo, cada sequência s foi representada por 29 atributos globais e intrínsecos, como descrito em Loong e Mishra 2007 e Loong e Mishra 2006.

Definindo $L(s)$ como o tamanho de s, os atributos foram obtidos como segue:

Frequência de dinucleotídeo – $\%XY$ tal que $X, Y \in \sum = [A, C, G, U]$, calculada por:

$$\%XY(s) = \frac{F_{XY}(s)}{L(s)} \times 100, \tag{4.1}$$

onde $F_{XY}(s)$ representa a frequência absoluta do dinucleotídeo XY em s.

Frequência de dinucleotídeo agregado – taxa $\%(G + C)$, calculada por:

$$Ratio_{GC}(s) = \frac{F_G(s) + F_C(s)}{L(s)} \times 100, \tag{4.2}$$

onde $F_G(s)$ e $F_C(s)$ representam a frequência absoluta de G e C em s respectivamente.

[5] http://deepbase.sysu.edu.cn/

[6] Obtido em ftp://ftp.jgi-psf.org/pub/JGI_data/phytozome/v5.0/Gmax/assembly/sequences/Gmax.main_genome.scaffolds.fasta.gz

[7] www.phytozome.net

[8] www.phytozome.net/biomart/martview

Mínima energia livre de dobramento ajustada (Adjusted Minimal Folding Free Energy - AMFE) (Zhang et al. 2006) – representa a energia livre de dobramento mínima (*Minimal Folding Free Energy - MFE*) em kcal/mol ajustada para representar uma sequência de 100 nucleotídeos, calculada por:

$$AMFE(s) = \frac{MFE(s)}{L(s)} \times 100, \tag{4.3}$$

onde a MFE foi obtida através da biblioteca RNAlib[9] a qual pertence ao pacote Vienna RNA(Hofacker et al. 1994), uma implementação em ANSI C do algoritmo de Zuker. Este algoritmo realiza uma avaliação completa de todas as possíveis estruturas secundárias formadas a partir de uma sequência RNA e determina aquela que possui a menor energia livre (Reedera et al. 2006).

Índice MFE 1 (MFE Index 1 - $\mathbf{MFEI_1}$) (Zhang et al. 2006) – razão entre $AMFE(s)$ e $Ratio_{GC}(s)$ calculada por:

$$MFEI_1(s) = \frac{AMFE(s)}{Ratio_{GC}(s)}. \tag{4.4}$$

Índice MFE 2 (MFE Index 2 - $\mathbf{MFEI_2}$) (Loong e Mishra 2006) – razão entre a $MFE(s)$ e o número de caules na estrutura secundária, que são motifs estruturais formados por mais de dois pares de base em sequência, calculado por:

$$MFEI_2(s) = \frac{AMFE(s)}{stem(db_s)}, \tag{4.5}$$

onde a função $stem()$ recebe como entrada a estrutura secundária de s na notação ponto-parêntesis (*dot-bracket*) db_s, obtida pela biblioteca RNAlib, e retorna o número total de caules.

Propensão do pareamento de bases ajustada (P) (Schultes, Hraber e LaBean 1999) – número total dos pares de base normalizado por $L(s)$, obtido através de:

$$P(s) = \frac{BP(s)}{L(s)}, \tag{4.6}$$

onde $BP(s)$, calculada pela biblioteca RNAlib, representa o número de pares de base na estrutura secundária de s.

Distância do pareamento de bases ajustada (D) (Freyhult, Gardner e Moulton 2005) e **Entropia de Shannon normalizada (Q)** (Gruber et al. 2008) – calculadas pelas Equações (4.7) e (4.8) respectivamente, onde a matriz ρ_{ij}, obtida pela biblioteca RNAlib através

[9] `http://www.tbi.univie.ac.at/~ivo/RNA/`

do algoritmo de McCaskill, representa a probabilidade das bases s_i e s_j formarem par.

$$D(s) = \frac{\sum_{i<j}(\rho_{ij} - \rho_{ij}^2)}{L(s)} \tag{4.7}$$

$$Q(s) = \frac{-\sum_{i<j}\rho_{ij} log_2(\rho_{ij})}{L(s)} \tag{4.8}$$

Grau de compacticidade (F), o qual caracteriza a complexidade da topologia do RNA (Gan et al. 2004) – calculado por uma implementação do algoritmo RNAspectral (Loong e Mishra 2006). Este algoritmo recebe como entrada db_s, obtida pela biblioteca RNAlib, cria a representação da estrutura secundária do RNA como um grafo planar em árvore e então calcula sua matriz Laplaciana, de modo que tal matriz é a diferença entre as matrizes de grau e de adjacência do grafo. O grau de compacticidade é o segundo menor valor diferente de zero pertencente ao conjunto de autovalores da matriz Laplaciana do grafo.

Os últimos 5 atributos foram baseados no trabalho de (Loong e Mishra 2006) onde os autores afirmam, mesmo considerando a pequena quantidade de dados utilizada (1806 pre-miRNAs conhecidos, 12.387 ncRNAs e 31 mRNAs) que os escores Z dos atributos AMFE, P, D, Q e F são estatisticamente capazes de diferenciar sequências pre-miRNA de outros tipos de ncRNAs e mRNAs. Os escores Z desses atributos são calculados por:

$$ZScore(A) = \frac{A_s - \mu(A_{shuffled})}{\sigma(A_{shuffled})}, \tag{4.9}$$

onde A_s representa o valor do atributo A da sequência s, $A_{shuffled}$ é um vetor dos valores do atributo A, calculados para cada uma das 10^4 sequências geradas pelo embaralhamento de dinucleotídeos em s, a função $\mu()$ retorna a média aritmética e a função $\sigma()$ retorna o desvio padrão. Deste modo, as 29 características estão completas.

4.2.3 Considerações Sobre a Implementação e o Ambiente de Execução dos Programas

Nesta Seção serão descritos os detalhes sobre a implementação e do ambiente de execução dos *Scripts* desenvolvidos neste trabalho.

Todos os *scripts* foram escritos na linguagem de programação Python[10] devido à sua facilidade para manipular variáveis de texto e o módulo Swig [11] foi utilizado para criar a interface entre a biblioteca RNAlib, escrita em linguagem C, e os *scripts* 2, 3 e 4.

Para a execução de todos os *scripts* desenvolvidos neste trabalho, foram

[10] `www.python.org`
[11] `www.swig.org`

utilizadas duas máquinas paralelas, no formato multiprocessador, conforme discutido na Seção 3.3. A arquitetura foi composta por: *i)* Sun Fire 4150 com dois processadores Intel Quad Core®, totalizando 8 núcleos de processamento com 32 GB de memória RAM e; *ii)* Sun Fire X4450 com dois processadores Intel Six Core®, totalizando 12 núcleos de processamento com 32 GB de memória RAM. Ambas possuem arquitetura 64 bits e utilizam como Sistema Operacional a distribuição CentOS[12] versão 5.4 do Linux.

O módulo Parallel Python[13] foi utilizado durante as implementações do *script* 4 e do algoritmo para treinamento da SVM, a fim de utilizar os recursos de computação paralela disponíveis na arquitetura de execução.

4.2.4 Pré-processamento dos Dados

Esta Seção descreve os *scripts* desenvolvidos para o pré-processamento dos dados obtidos, descritos na Seção 4.2.1. Os *scripts* 1-3 são utilizados para estabelecer as sequências candidatas e o *Script* 4 calcula os atributos descritos na Seção 4.2.2 para cada sequência.

A Figura 4.2 ilustra a metodologia empregada para preparação dos dados referentes a esta etapa do trabalho, desde a obtenção das sequências pertencentes aos conjuntos positivo, negativo e pre-miRNA candidatos, seu pré-processamento realizado pelos *Scripts* 1-4 até a gravação dos valores dos atributos no banco de dados.

4.2.4.1 Script 1

O primeiro *script* recebe como entrada os transcritos da soja no formato FASTA. O formato FASTA é muito utilizado para a gravação de sequências em arquivos, pois é simples e flexível (Gibas e Jambeck 2001). Cada registro é representado por uma linha iniciada com o carácter '>' (maior que) seguido por um ou mais identificadores da sequência. O restante do registro contém a sequência em si, conforme pode ser observado no exemplo da Figura 4.3.

Em sua saída, o *script* 1 grava apenas as sequências referentes aos introns de cada transcrito em um novo arquivo, também no formato fasta.

4.2.4.2 Script 2

Este *script* tem como tarefa varrer as regiões intergênicas do genoma em busca de sequências candidatas a pre-miRNA. As sequências desejadas são aquelas que podem

[12] `www.centos.org`
[13] `http://www.parallelpython.com/`

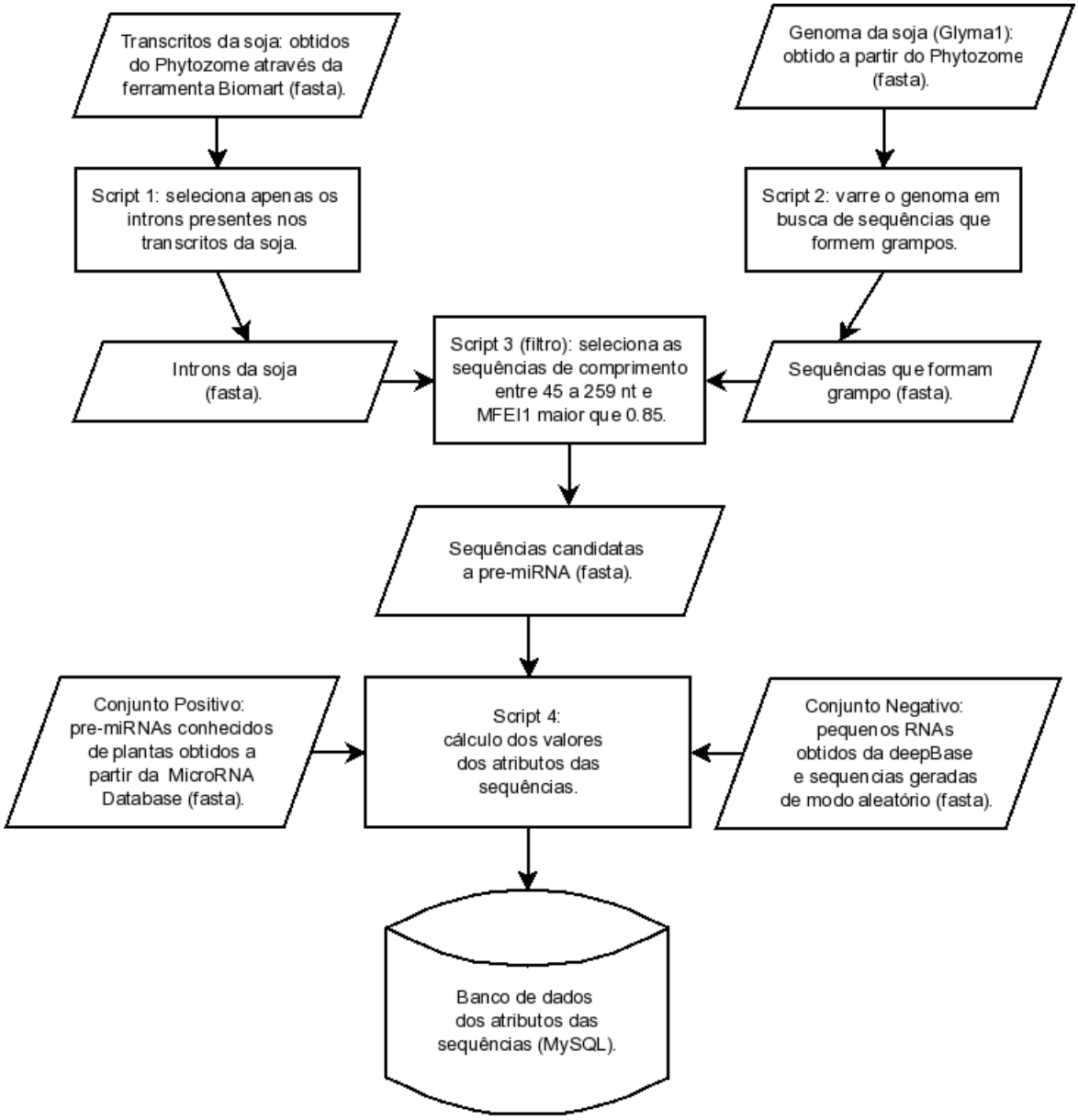

Figura 4.2: As sequências são obtidas de diferentes fontes e, após o cálculo do valor de seus atributos, estes são centralizados no banco de dados de atributos das sequências.

```
>id1_01|id1_02|...|id1_n
UACCCCCCGUGUGUUUCGAACAACGGCGGACGGUUCUGGUGCCUCUACCACGUGAUUCAAUUCGAGGACU
CCGCGAGCAUCAAUGGCGAACUCAAUGACGCGCCACUAUGUCACUGUCGAAGUUAGUCUCCAAAUCUUCA
CCUUUGCUUUCGAUGAUUUUGUCCUUCCUAAUCCUAAACCUAUGGAACCCUGGCGACGCCAAUGCCGCGG
UCUUCCUUCAUACGCCCUCACUUUUCUGAAUAGGGGUCGUAGCAUCAGUGCUAUGUCUUGGAUACGAUUG
CUAGUAUGUCUCAUAUUAGUCCAGUCAGCAAUCCUCGCUUUC
>id2_01|id2_02|...|id2_n
UUAAUGUUUGCCCGUGUCCUUGUACGGUUUAGUAAACAUCGUAGAAUGGACAAACAGUUUUUGACAGAAG
UCUAGGCAGGGCUAGGCGCUGUGCCAGCCGCACGACUUUGCGCCCAAGUCCUAGAAUAGGAUGUUUACCA
UACGCUUAACGUCUACAUUGCCCGUCACGUCGAGAUUGUCACCUGUCCGAGACUGUUUGGCUCGGUCUGU
CGAACGCGAAGUGUAACAGGCGUAGCACGAAAGAGCGUCAAAAGACAUGAAUACGUCAAACACAGAAGUU
CGGGAACA
```

Figura 4.3: Exemplo de um arquivo no formato FASTA, com dois registros.

formar uma estrutura de grampo, através de dobra e pareamento consigo mesma.

Como critério para selecionar este tipo de sequência candidata, foi determinada uma janela de 35 nts (braço 1), um espaço de, no mínimo, cinco e no máximo 105 nts (volta) e outros 35 nts (braço 2). Enquanto o espaço entre os braços 1 e 2 não ultrapassar 105 nts, ou seja, enquanto a volta for menor que 105 nts, a sequência do braço 1 é comparada com o reverso complementar da sequência do braço 2 através da Distância de Hamming(Hamming 1950).

A Distância de Hamming considera cada posição entre as sequências comparadas, marcando com o valor 0 se o nucleotídeo do braço 2 é o reverso complementar do nucleotídeo presente na sequência do braço 1 ou, caso contrário, marcando o valor 1. A Distância de Hamming entre as sequências é a soma de todos os valores marcados durante a comparação.

Se a Distância de Hamming resultante desta comparação for menor que 10, a sequência completa é escolhida como candidata a pre-miRNA e o algoritmo continua a partir do primeiro nucleotídeo após a sequência selecionada.

4.2.4.3 Script 3

O *Script* 3 recebe como entrada os arquivos de saída dos *Scripts* 1-2 e executa dois filtros sobre os pre-miRNA candidatos, com a finalidade de excluir aquelas sequências que provavelmente não possuem miRNA: $44 \leq L(s) \leq 259$ nts (Zhang, Pan e Stellwag 2008) e $\text{MFEI}_1 > 0.85$ (Zhang et al. 2006). As sequências resultantes desta etapa são gravadas em um novo arquivo, também no formato fasta.

Após o uso dos *scripts* 1–3, obtivemos um total de 1.205 sequências candidatas a pre-miRNA originárias dos introns da soja e 101.326 pre-miRNA candidatos de regiões intergênicas do genoma da soja.

4.2.4.4 Script 4

O *Script* 4 recebe como entrada um arquivo no formato fasta, contendo as sequências pertencentes aos conjuntos positivo, negativo e candidatas a pre-miRNA. Em seguida, calcula os 29 atributos de cada sequência, com o uso da biblioteca RNAlib e do algoritmo RNAspectral e, finalmente, grava seus valores em um banco de dados MySQL[14], chamado banco de dados de valores dos atributos das sequências. A Figura 4.4 mostra a visão explodida deste *script*.

No banco de dados de valores dos atributos das sequências, os conjuntos

[14] `www.mysql.com`

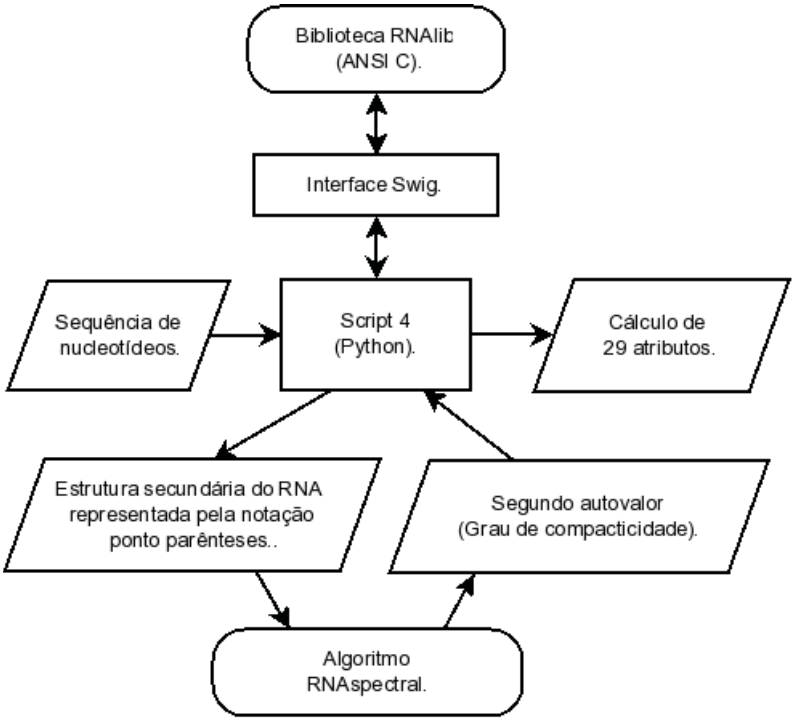

Figura 4.4: Visão explodida do *Script* 4, o qual calcula os 29 atributos para cada sequência utilizada no trabalho.

positivo, negativo e pre-miRNA candidatos são separados por valor esperado, do seguinte modo: sequências com valor esperado igual a "1" pertencem ao conjunto positivo; sequências com valor esperado igual a "-1" representam o conjunto negativo; as sequências candidatas a pre-miRNA possuem valor esperado igual a "0". Deste modo, as sequências e os valores de seus atributos ficam disponíveis para uso futuro.

Devido ao volume de processamento necessário ao cálculo dos atributos, a implementação do *Script* 4 foi realizada de modo que sua execução ocorra em paralelo, do seguinte modo: um processo foi criado para cada sequência a ser analisada. Cada processo foi inserido em uma fila e, conforme existisse um núcleo de processamento livre, um processo é retirado da fila e alocado a este núcleo para execução.

4.2.5 Aprendizagem com SVM

As etapas de treinamento e teste são necessárias para que o algoritmo SVM aprenda sobre o problema a ser resolvido e, consequentemente, realize a classificação das sequências candidatas de forma correta. Deste modo, o conjunto de treinamento (CT) foi construído com 400 sequências, como segue: 131 pre-miRNAs da soja (*Glycine max*) e 69 pre-miRNAs de *Arabidopsis thaliana*, retirados do conjunto positivo. Do conjunto negativo, foram selecionadas 175 sequências de snoRNAs de *Arabidopsis thaliana* mais 25 sequências aleatórias.

Adicionalmente, foi criado um conjunto de dados para a validação do aprendizado (CV), composto por 200 sequências, das quais 100 originárias do conjunto positivo, escolhidas aleatoriamente entre os pre-miRNAs de *Medicago truncatula* e 100 sequências aleatórias obtidas a partir do conjunto negativo. As sequências presentes no conjunto CV não pertencem ao conjunto CT, de modo que as sequências utilizadas na etapa de treinamento não participam da validação do aprendizado.

Todos os dados apresentados ao algoritmo SVM foram normalizados por atributo, no intervalo $[-1.0, 1.0]$ pela Equação 4.10, onde X é um vetor contendo os valores que serão normalizados, X_i representa o i-ésimo valor de X, X_{max} e X_{min} são o maior e o menor valor em X, respectivamente. A Figura 4.5 mostra dois exemplos de vetores de dados apresentados como entrada à SVM.

$$\hat{X}_i = 2 \times \frac{X_i - X_{min}}{X_{max} - X_{min}} - 1 \tag{4.10}$$

%AA	...	%UU	RatioGC	AMFE	MFEI1	MFEI2	P	D	Q	F	zAMFE	zP	zD	zQ	zF	Valor esperado
-0.13	...	0.39	-0.28	0.01	0.21	-0.12	0.11	1	0.14	-0.04	0.09	0.55	-0.15	-0.68	-0.31	-1
0.19	...	-0.85	1	0.63	0.01	-0.95	-0.76	0.45	-0.51	-0.34	0.16	0.49	-0.23	-0.10	0.33	1

Figura 4.5: Exemplo do formato dos dados apresentados ao algoritmo SVM após o processo de normalização, que ajusta os valores para o intervalo $[-1.0, 1.0]$.

Conforme discutido na Seção 3.2.1, o treinamento do algoritmo SVM necessita de dois parâmetros: i) σ^2 para a função núcleo e, ii) constante de regularização C para o algoritmo SMO. O par (σ^2, C) deve ser escolhido de modo a maximizar a exatidão ($exat$), a sensibilidade ($sens$) e a especificidade ($espec$) do classificador, as quais são calculadas pelas Equações 4.11–4.13, onde VP, VN, FP e FN denotam o número de resultados verdadeiros positivos, verdadeiros negativos, falsos positivos e falsos negativos respectivamente.

$$exat = \frac{VP + VN}{VP + VN + FP + FN} \tag{4.11}$$

$$sens = \frac{VP}{VP + FN} \times 100 \tag{4.12}$$

$$espec = \frac{VN}{VN + FP} \times 100 \tag{4.13}$$

A etapa de treinamento do algoritmo foi realizada conforme proposto por (Hsu, Chang e Lin 2010), onde 100 possibilidades para o par (C, σ^2) são testadas, como segue: $(C_i, \sigma_j^2) \mid C_i, \sigma_j^2 \in \{2^{-4}, 2^{-3}, ..., 2^4, 2^5\}$ e utilizado o método k-fold, onde $k = 5$. Deste modo, o conjunto de treinamento foi dividido em cinco partes iguais, quatro das quais utilizadas para

treinamento (80% de TS ou 320 sequências) e uma para teste (20% de TS ou 80 sequências).

O treinamento/teste é realizado e o processo repetido até que todas as partes sejam usadas para o teste. Os resultados finais para a exatidão, sensibilidade e especificidade são obtidos através da média entre os k resultados. A Figura 4.6 mostra a divisão dos dados em conjuntos de treinamento e teste, utilizada no método $5-fold$, com a finalidade de treinar o algoritmo SVM e obter o melhor par (σ^2, C).

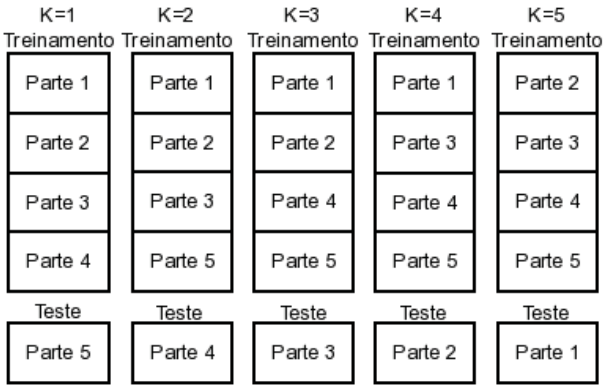

Figura 4.6: Divisão dos dados pelo método $5-fold$ para o treinamento/teste do algoritmo SVM, utilizado a fim de obter o melhor par (σ^2, C).

O método $5-fold$ foi realizado por um *script* desenvolvido em paralelo, no qual foram criados cinco processos, um para cada iteração do método.

Após a obtenção do melhor par (C, σ^2), o algoritmo SVM foi treinado uma última vez, com o uso do conjunto CT e sem o método $5-fold$. Em seguida, o aprendizado foi validado pelo conjunto CV e, finalmente, utilizado para classificar as sequências candidatas, conforme ilustrado na Figura 4.7, que mostra também a etapa de verificação de similaridade das sequências classificadas como pre-miRNA com sequências que codificam proteína presentes no genoma da soja.

4.2.6 Verificação das Sequências Classificadas como Pre-miRNA

Em plantas, os miRNAs geralmente possuem complementariedade quase perfeita com o seus alvos (Bentwich 2005). Portanto, a verificação computacional das sequências classificadas pelo algoritmo SVM como pre-miRNA e, consequentemente, a predição dos genes influenciados pelo miRNA, pode ser realizada através de um algoritmo de busca por similaridade.

Nesta etapa, foi utilizado o BLAST[15] (Altschul et al. 1990), de modo que a busca por similaridade foi realizada através do algoritmo blastn (nucleotídeo x nucleotí-

[15] `ftp://ftp.ncbi.nih.gov/blast/`

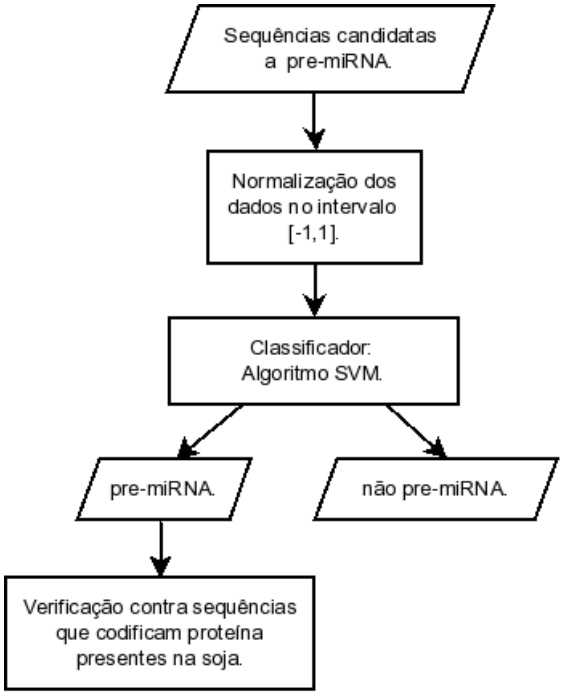

Figura 4.7: Classificação das sequências candidatas e posterior verificação, no genoma da soja, daquelas classificadas como pre-miRNA.

deo) contra sequências de DNA complementar (cDNA) da soja (*Glycine max*), obtidas do Phythozome (Schmutz et al. 2010), já que o cDNA contém apenas sequências codificadoras de proteínas.

Uma vez que as sequências candidatas a pre-miRNA utilizadas possuem $45-259$ nts de comprimento e o miRNA maduro possui $18-22$ nt, a opção tamanho da palavra (-W) foi definida com o valor 18, estabelecendo que os acertos (*hits*) entre as sequências pre-miRNA e as sequências cDNA possuam, no mínimo, 18 nucleotídeos. O valor esperado (E-Value), que indica a probabilidade de um alinhamento ocorrer ao acaso, foi deixado com o valor padrão (10).

5 RESULTADOS

Este Capítulo apresenta os resultados obtidos através da busca por homologia de pre-miRNAs conhecidos em outras espécies de plantas, presentes no genoma da soja (Seção 5.1). Mostra, também, o classificador obtido através do treinamento do algoritmo SVM, bem como as sequências candidatas por ele classificadas como possíveis pre-miRNAs e seus prováveis alvos (Seção 5.2).

5.1 Busca por Homologia

A Tabela 5.1 apresenta os resultados obtidos através do alinhamento das sequências de plantas contra o genoma da soja, através do algoritmo SOAPaligner/SOAP2. Na coluna 'Id. sequência' estão os identificadores das sequências de pre-miRNA, a coluna 'Espécie' mostra a espécie a qual o pre-miRNA pertence, a coluna 'Sentido' identifica o sentido no qual ocorreu o alinhamento, a coluna 'Id. ref.' mostra o identificador do cromossomo no qual ocorreu o alinhamento, a coluna 'Posição' diz onde o alinhamento foi iniciado, nesse cromossomo, a coluna 'Tamanho' apresenta o tamanho da sequência de pre-miRNA identificada e a coluna 'Hits' diz a quantidade de bases que foram alinhadas.

Tabela 5.1: Resultados obtidos através da busca por homologia.

Id. sequência	Espécie	Sentido	Id. ref.	Posição	Tamanho	Hits
peu-MIR2910	*Populus euphratica*	$3' \rightarrow 5'$	Gm13	15242270	59	59
peu-MIR2914	*Populus euphratica*	$3' \rightarrow 5'$	Gm13	15058225	63	62
vun-MIR2118	*Vigna unguiculata*	$5' \rightarrow 3'$	Gm20	35349747	129	129

Com base nos resultados, nota-se que duas sequências de pre-miRNA obtidas da *Populus euphratica* alinharam com o genoma da soja, mais precisamente em seu cromossomo 13. A sequência identificada por peu-MIR2914 apresentou uma diferença (*mismatch*), enquanto que a sequência peu-MIR2910 realizou pareamento para todas as suas bases. Ambas as sequências alinharam no sentido $3' \rightarrow 5'$.

A sequência identificada por vun-MIR2118, pertencente à *Vigna unguiculata* alinhou em uma região do cromossomo 20 da soja, no sentido $5' \rightarrow 3'$ de modo que as suas

129 bases realizaram pareamento.

5.2 Busca com o Uso de Aprendizagem de Máquina

Esta Seção apresenta o classificador desenvolvido com o uso do algoritmo SVM, mostrando os valores obtidos quanto à exatidão, sensibilidade e especificidade nas etapas de treinamento/teste e validação (Seção 5.2.1) e, em seguida, mostra os pre-miRNAs identificados com o uso deste classificador (Seção 5.2.2).

5.2.1 Classificador SVM

Os valores $C = 1,0$ e $\sigma^2 = 1,0$ apresentaram maior exatidão, sensibilidade e especificidade após os 100 primeiros testes. Em seguida, com a verificação nas proximidades, os valores $C = 1,5$ e $\sigma^2 = 1,1$ aumentaram a exatidão e a especificidade do classificador, com leve queda em sua sensibilidade. Portanto, esses valores foram escolhidos como definitivos, o algoritmo foi novamente treinado com o conjunto TS, sem o uso do método *5-fold*, e passou por validação através do conjunto VS.

A Tabela 5.2 mostra os resultados obtidos pelo classificador quanto à exatidão, sensibilidade e especificidade, nas etapas de treinamento/teste e validação.

Tabela 5.2: Resultados apresentados pelo algoritmo SVM nas fases de treinamento/teste e validação.

Dados de treinamento	C	σ^2	Exatidão	Sensibilidade	Especificidade
CT (com *k-fold*)	$1,0$	$1,0$	$0,92$	$91,34\%$	$93,77\%$
CT (com *k-fold*)	$1,5$	$1,1$	$0,93$	$91,30\%$	$94,84\%$
CT (sem *k-fold*) e validação por CV	$1,5$	$1,1$	$0,92$	$89,0\%$	$95,0\%$

5.2.2 Pre-miRNAs Preditos e Seus Prováveis Alvos

A Tabela 5.3 mostra de, modo parcial[1], as sequências candidatas classificadas como pre-miRNA pelo algoritmo SVM e seus prováveis alvos no genoma da soja e identificados através da verificação realizada com o algoritmo BLAST. Na coluna 'Identificador' estão os identificadores criados para cada sequência candidata, com base no transcrito do qual ela foi retirada. A coluna 'Região' identifica de qual região a sequência candidata foi retirada e a

[1] A tabela completa pode ser observada no Apêndice A

coluna 'Alvo(s)' apresenta transcritos da soja com os quais a sequência candidata realizou alinhamento através da verificação com o uso do algoritmo BLAST.

Tabela 5.3: Pre-miRNAs preditos e seus prováveis alvos (parcial).

Identificador	**Região**	**Alvo(s)**
Glyma0041s00200.1(1)	Intrônica	Glyma13g27410.1
Glyma01g09430.1(1)	Intrônica	Glyma02g23640.1, Glyma05g21210.1, Glyma20g20840.1.
Glyma01g22490.1(1)	Intrônica	Glyma14g33280.1.
Glyma01g28310.1(1)	Intrônica	Glyma06g03190.3, Glyma17g08030.1.
Glyma01g28880.1(1)	Intrônica	Glyma10g01560.1.
Glyma01g37140.1(1)	Intrônica	Glyma08g37270.1.
Glyma02g01820.1(1)	Intrônica	Glyma10g01890.1.
Glyma02g03610.1(1)	Intrônica	Glyma15g08560.1.
Glyma02g07110.1(1)	Intrônica	Glyma18g15920.1.
Glyma02g09970.1(1)	Intrônica	Glyma06g41100.1, Glyma06g41050.1, Glyma14g38810.1, Glyma02g40500.1.
Glyma02g15950.1(1)	Intrônica	Glyma09g24270.1, Glyma10g12900.1, Glyma07g00350.1.
Glyma02g16370.1(1)	Intrônica	Glyma17g02540.1, Glyma07g38180.1, Glyma17g02540.2.
Glyma02g23640.1(1)	Intrônica	Glyma02g23640.1, Glyma01g20840.1, Glyma14g32230.1, Glyma05g21210.1, Glyma01g09430.1, Glyma20g20840.1, Glyma09g23710.1, Glyma02g33010.1, Glyma07g34540.2, Glyma15g33010.1, Glyma10g22610.1.

Ao todo, 160 sequências candidatas preditas como pre-miRNA pelo classificador apresentaram alinhamento com pelo menos um transcrito presente no genoma da soja. Tais sequências são indicadas na Tabela A.1, presente no Apêndice A.

Essas sequências candidatas tiveram origem em regiões intrônicas do genoma da soja. Segundo Brown, Marshall e Echeverria 2008, até o início do ano de 2008, existiam apenas 11 pre-miRNAs identificados a partir de introns de plantas. Logo, os resultados obtidos apontam a provável existência de pre-miRNAs em introns da soja.

Apesar da importância econômica da soja, conforme discutido no Capítulo 1, não foi encontrada na literatura, a aplicação de métodos de aprendizagem de máquina para a identificação de pre-miRNAs em seu genoma, para a comparação dos resultados. Este fato pode ser explicado pelo caráter recente da pesquisa de miRNA e também pelo genoma da soja ainda não estar completamente sequenciado.

Observando os resultados obtidos no trabalho de Sewer et al. 2005, que também utilizou o algoritmo SVM e foi aplicado em três genomas animais distintos, resultando

na predição de 224, 192 e 208 pre-miRNAs (contando as sequências conhecidas), conforme observado na Seção 1.1.2, a quantidade de sequências preditas neste trabalho (160) pode-se considerar próxima ao esperado.

Finalmente, análises experimentais em laboratório devem ser realizadas sobre os resultados obtidos por esta etapa do trabalho, a fim de confirmar a hipótese de que estas sequências são realmente pre-miRNAs e, caso positivo, verificar o miRNA maduro contido em cada pre-miRNA.

6 CONCLUSÃO

Metodologias *in silico* (totalmente desenvolvidas em computador), como a utilizada aqui, possuem alguns fatores positivos que justificam o seu uso: i) aumentam a velocidade de análise dos dados e, consequentemente, possibilitam a avaliação de um volume maior de informações; ii) os resultados podem indicar regiões no genoma, as quais possuem maior probabilidade de confirmação experimental, da hipótese estudada.

Conforme proposto, foram utilizadas abordagens computacionais a fim de identificar pre-miRNAs no genoma da soja (*Glycine max*). Nesse aspecto, foram utilizadas a busca por homologia e técnicas de aprendizagem de máquina.

Logo, como contribuições deste trabalho, destacam-se:

- A identificação, no genoma da soja, de três pre-miRNAs existentes de duas espécies de plantas (dois de *Populus euphratica* e um de *Vigna unguiculata*) através do uso de busca por homologia;
- O desenvolvimento de um classificador SVM direcionado para a predição de pre-miRNAs em plantas;
- A predição de 160 prováveis pre-miRNAs no genoma da soja;
- A possível existência de miRNAs nos introns da soja.

6.1 Trabalhos futuros

Como trabalhos futuros, uma metodologia será desenvolvida, a fim de estabelecer o miRNA presente nas sequências pre-miRNA. Os resultados obtidos serão validados experimentalmente. Um estudo será realizado com os genes alvos dos miRNAs identificados e o conhecimento obtido será aplicado para o desenvolvimento de cultivares de soja mais produtivas, de modo que estas sejam melhor adaptadas às variações de clima e solo existentes no Brasil e também possuam maior resistência à pragas e doenças.

REFERÊNCIAS

Ales 2008 ALES, V. T. O algoritmo sequential minimal optimisation para resolução do problema de support vector machine: Uma técnica para reconhecimento de padrões. Dissertação (Mestrado) — Universidade Federal do Paraná, Curitiba, 2008.

Altschul et al. 1990 ALTSCHUL, S. et al. Basic local alignment search tool. *J. Mol. Biol.*, v. 215, p. 403–410, 1990.

Bentwich 2005 BENTWICH, I. Prediction and validation of microrna and their targets. *FEBS Letters*, v. 579, p. 5904–5910, 2005.

Bonnet et al. 2004 BONNET, E. et al. Detection of 91 potential conserved plant microrna in arabidopsis thaliana and oryza sativa identifies important target genes. *Proceedings of the National Academy of Sciences of the United States of America*, v. 101, n. 31, p. 11511–11516, Agosto 2004.

Brown, Marshall e Echeverria 2008 BROWN, J. W. S.; MARSHALL, D. F.; ECHEVERRIA, M. Intronic noncoding rnas and splicing. *Trends in Plant Science*, v. 13, n. 7, p. 335–342, June 2008.

Brown 1999 BROWN, T. A. *Genomes*. Oxford, UK: Bios Scientific Publishers, 1999.

Bruggeman et al. 2007 BRUGGEMAN, F. J. et al. Introdution to systems biology. In: BAGINSKY, S.; FERNIE, A. R. (Ed.). Switzerland: Birkhaüser Verlag, 2007. EXS 97, p. 1–20.

Burges 1998 BURGES, C. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, n. 2, p. 121–167, 1998.

Cáceres, Mongelli e Song 2001 CÁCERES, E. N.; MONGELLI, H.; SONG, S. W. Algoritmos paralelos usando cgm/pvm/mpi: Uma introdução. *Anais do XXI Congresso da Sociedade Brasileira de Computação*, v. 2, p. 219–278, 2001.

Conab 2010 CONAB. *Acompanhamento da safra brasileira: grãos, oitavo levantamento*. Brasília, Maio 2010. Disponível em: <http://www.conab.gov.br/conabweb/download/safra-/8graos_6.5.10.pdf>.

Cox e Miller 1977 COX, D. R.; MILLER, H. D. *Theory of Stochastic Processes*. 1. ed. New York: Halsted Press, 1977.

Duret e Abdeddaim 2000 DURET, L.; ABDEDDAIM, S. Multiple alignments for structural, functional, or phylogenetic analysis of homologous sequences. In: HIGGINS, D.; TAYLOR, W. (Ed.). *Bioinformatics: Sequence, structure and data banks*. New York: Oxford University Press, 2000.

Feller 1968 FELLER, W. *A Introdution to Probability Theory and its Applications*. 3. ed. New York: Wiley International, 1968.

Freyhult, Gardner e Moulton 2005 FREYHULT, E.; GARDNER, P.; MOULTON, V. A comparison of rna folding measures. *BMC Bioinformatics*, v. 6, n. 1, p. 241+, Outubro 2005.

Gan et al. 2004 GAN, H. H. et al. Rag: Rna-as-graphs database—concepts, analysis, and features. *Bioinformatics*, v. 20, n. 8, p. 1285–1291, Fevereiro 2004.

Ghosh, Chakrabarti e Mallick 2007 GHOSH, Z.; CHAKRABARTI, J.; MALLICK, B. mirnomics - the bioinformatics of microrna genes. *Biochemical and Biophysical Research Communications*, v. 363, p. 6–11, 2007.

Gibas e Jambeck 2001 GIBAS, C.; JAMBECK, P. *Desenvolvendo Bioinformática*: Ferramentas de *software* para aplicações em biologia. Rio de Janeiro: Campus, 2001.

Graur e Li 2000 GRAUR, D.; LI, W.-H. *Fundamentals of Molecular Evolution*. Sunderland: Sinauer Associates, 2000.

Gruber et al. 2008 GRUBER, A. et al. Strategies for measuring evolutionary conservation of rna secondary structures. *BMC Bioinformatics*, v. 9, n. 1, p. 122+, Fevereiro 2008.

Hamming 1950 HAMMING, R. W. Error detecting and error correcting codes. *Bell System Technical Journal*, v. 29, n. 2, p. 147–160, 1950.

Haykin 1999 HAYKIN, S. *Redes Neurais: princípios e práticas*. Porto Alegre: Bookman, 1999.

Hertel e Stadler 2006 HERTEL, J.; STADLER, P. F. Hairpins in a haystack: recognizing microrna precursors in comparative genomics data. *Bioinformatics*, v. 22, n. 14, p. e197–e202, 2006.

Higgs e Attwood 2005 HIGGS, P. G.; ATTWOOD, T. K. *Bioinformatics and Molecular Evolution*. Malden: Blackwell Science, 2005.

Hofacker et al. 1994 HOFACKER, I. L. et al. Fast folding and comparison of rna secondary structures. *Monatshefte fúr Chemie*, v. 125, p. 167–188, 1994.

Hou, Ying e Li 2008 HOU, Y.; YING, X.; LI, W. Computational approaches to microrna discovery. *Yi Chuan*, 2008.

Hsu, Chang e Lin 2010 HSU, C.-W.; CHANG, C.-C.; LIN, C.-J. *A Practical Guide to Support Vector Classification*. Abril 2010.

Jiang et al. 2007 JIANG, P. et al. Mipred: classification of real and pseudo microrna precursors using random forest prediction model with combined features. *Nucleic Acids Research*, v. 35, n. Suppl 2, p. W339–W344, 2007.

Jones-Rhoades e Bartel 2004 JONES-RHOADES, M. W.; BARTEL, D. P. Computational identification of plant micrornas and their targets, including a stress-induced mirna. *Molecular Cell*, v. 14, p. 787–799, 2004.

Li et al. 2008 LI, R. et al. Soap: Short oligonucleotide alignment program. *Bioinformatics*, v. 24, n. 5, p. 713–714, 2008.

Loong e Mishra 2006 LOONG, S. N. K.; MISHRA, S. K. Unique folding of precursor micrornas: Quantitative evidence and implications for de novo identification. *RNA*, v. 13, p. 170–187, 2006.

Loong e Mishra 2006 LOONG, S. N. K.; MISHRA, S. K. Unique folding of precursor micrornas: Quantitative evidence and implications for de novo identification. *RNA Supplementary materials*, v. 13, 2006.

Loong e Mishra 2007 LOONG, S. N. K.; MISHRA, S. K. De novo svm classification of precursor micrornas from genomic pseudo hairpins using global and intrinsic folding measures. *Structural Bioinformatics*, v. 23, n. 11, p. 1321–1330, Janeiro 2007.

Lorena e Carvalho 2007 LORENA, A. C.; CARVALHO, A. C. P. L. F. d. Uma introdução às support vector machines. *RITA*, XIV, n. 2, p. 44–67, 2007.

Mendes, Freitas e Sagot 2009 MENDES, N. D.; FREITAS, A. T.; SAGOT, M.-F. Current tools for the identification of mirna genes and their targets. *Nucleic Acids Research*, v. 37, p. 2419–2433, 2009.

Nam et al. 2005 NAM, J.-W. et al. Human microrna prediction through a probabilistic co-learning model of sequence and structure. *Nucl. Acids Res.*, v. 33, n. 11, p. 3570–3581, Junho 2005.

Noble 2004 NOBLE, W. S. Support vector machine applications in computational biology. In: SCHOLKOPF, B.; TSUDA, K.; VERT, J.-P. (Ed.). *Kernel methods in computational biology*. Cambridge, MA: MIT Press, 2004. cap. 3.

Osuna, Freund e Girosi 1997 OSUNA, E.; FREUND, R.; GIROSI, F. Training support vector machines: An application to face detection. *Proccedings of CVPR'97*, Puerto Rico, p. 130–136, 1997.

Pasin e Kreutz 2003 PASIN, M.; KREUTZ, D. L. Arquitetura e administração de aglomerados. 3^a *Escola Regional de Alto Desempenho ERAD 2003*, p. 3–34, 2003.

Platt 1998 PLATT, J. C. *Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines*. [S.l.], Abril 1998.

Prosdocimi et al. 2003 PROSDOCIMI, F. et al. Bioinformática: Manual do usuário. *Biotecnologia Ciência e Desenvolvimento*, n. 29, p. 12–25, 2003.

Reedera et al. 2006 REEDERA, J. et al. Beyond mfold: Recent advances in rna bioinformatics. *Journal of Biotechnology*, v. 124, n. 1, p. 41–55, Junho 2006.

Rose e Navaux 2002 ROSE, C. A. F. de; NAVAUX, P. O. A. Fundamentos de processamento de alto desempenho. 2^a *Escola Regional de Alto Desempenho ERAD 2002*, São Leopoldo, RS, v. 1, p. 3–29, 2002.

Rose e Navaux 2003 ROSE, C. A. F. de; NAVAUX, P. O. A. *Arquiteturas Paralelas*. 1. ed. Porto Alegre, RS: Editora Sagra Luzzatto, 2003.

Schmutz et al. 2010 SCHMUTZ, J. et al. Genome sequence of the palaeopolyploid soybean. *Nature*, v. 463, p. 178–183, Janeiro 2010.

Schultes, Hraber e LaBean 1999 SCHULTES, E. A.; HRABER, P. T.; LABEAN, T. H. Estimating the contributions of selection and self-organization in rna secondary structure. *Journal of Molecular Evolution*, v. 49, n. 1, p. 76–83, Julho 1999.

Sewer et al. 2005 SEWER, A. et al. Identification of clustered microrna using an ab initio prediction method. *BMC Bioinformatics*, 2005.

Sunkar e Jagadeeswaran 2008 SUNKAR, R.; JAGADEESWARAN, G. In silico identification of conserved microrna in large number of diverse plant species. *BMC Plant Biology*, v. 8:37, 2008.

Xue et al. 2005 XUE, C. et al. Classification of real and pseudo microrna precursors using local structure-sequence features and support vector machine. *Bioinformatics*, v. 6, n. 310, Dezembro 2005.

Yang et al. 2010 YANG, J.-H. et al. deepbase: a database for deeply annotating and mining deep sequencing data. *Nucl. Acids Res.*, v. 38, n. suppl_1, p. D123–130, Janeiro 2010.

Yousef et al. 2006 YOUSEF, M. et al. Combining multi-species genomic data for microrna identification using naive bayes classifier. *Bioinformatics*, v. 22, n. 11, p. 1325–1334, 2006.

Yousef, Showe e Showe 2009 YOUSEF, M.; SHOWE, L.; SHOWE, M. A study of microrna in silico and in vivo: bioinformatics approaches to microrna discovery and target identification. *FEBS Journal*, v. 276, p. 2150–2156, 2009.

Zaha et al. 1996 ZAHA, A. et al. *Biologia Molecular Básica*. Porto Alegre: Mercado Aberto, 1996. (Ciência XXI).

Zhang et al. 2005 ZHANG, B. et al. Plant microrna: A small regulatory molecule with big impact. *Developmental Biology*, v. 289, p. 3–16, 2005.

Zhang, Pan e Stellwag 2008 ZHANG, B.; PAN, X.; STELLWAG, E. J. Identification of soybean microrna and their targets. *Planta*, v. 229, p. 161–182, 2008.

Zhang et al. 2006 ZHANG, B. H. et al. Evidence that mirnas are different from other rnas. *Cellular and Molecular Life Sciences*, v. 63, p. 246–254, 2006.

Zhang et al. 2010 ZHANG, Z. et al. Pmrd: plant microrna database. *Nucl. Acids Res.*, v. 38, n. suppl_1, p. D806–813, Janeiro 2010.

Zhao et al. 2010 ZHAO, D. et al. Pmirp: A pre-microrna prediction method based on structure-sequence hybrid features. *Artificial Intelligence in Medicine*, v. 49, n. 2, p. 127–132, 2010.

GLOSSÁRIO

Anti-senso: Sequência de DNA que é molde para síntese do RNA mensageiro.
Alinhamento de sequências: Modo de organizar duas sequências a fim de identificar regiões similares.
Aprendizagem supervisionada: Aprendizagem na qual um professor avalia os resultados obtidos pelo algoritmo durante seu treinamento.
Bias: Erro tendencioso ou sistemático em resultados estatísticos.
Braços do grampo: Partes do grampo (*hairpin*) antes e depois da volta (*loop*).
Caule: Região do grampo na qual ocorre o pareamento de duas ou mais bases em sequência.
Códon: Sequência de três nucleotídeos que codificam um aminoácido.
DNA complementar: Molécula de DNA obtida a partir de um mRNA, ou seja, pelo processo inverso de transcrição.
Formato FASTA: Formato de arquivo para armazenamento de sequências em modo texto. Detalhes na Seção 4.2.4.1.
Gene alvo: Gene no qual liga-se o miRNA maduro.
Grampo (hairpin): Formato da molécula de pre-miRNA. Ver Figura 4.1.
Junção (splicing): Processo que ocorre na molécula pri-miRNA a fim de originar o pre-miRNA. Detalhes na Seção 2.1.1.
Ligações fosfodiéster: Ligações químicas responsáveis pela estrutura das cadeias de DNA e RNA. Detalhes na Seção 2.1.
Micro RNA (miRNA): Pequena molécula de RNA que possui papel regulatório no organismo. Detalhes na Seção 2.1.1.
miRNA primário (pri-miRNA): Transcrito primário que contém o miRNA.
miRNA conservado: miRNA que existe em duas ou mais espécies.
miRNA maduro: Molécula do miRNA pronta para ligar-se ao complexo miRISC. Detalhes na Seção 2.1.1.
Mismatch: Não pareamento de bases em um alinhamento de sequências.
Ponte de hidrogênio: Ligação entre as cadeias de nucleotídeos da fita de DNA. Ver Figura 2.2.
Pós-transcricional: Momento após a transcrição do DNA e antes da tradução em proteína.
Região intergênica: Sequência de nucleotídeos que localizam-se nos intervalos entre os genes.
Ribossomo: Organela celular onde ocorre a síntese de proteínas.
RNA mensageiro (mRNA): Molécula de RNA que resulta do processo de transcrição do

DNA e serve como molde para a proteína, durante o processo de tradução. Detalhes na Seção 2.1.

RNA não codificante (ncRNA): Classe de RNA que não codifica proteína.

RNA ribossômico (rRNA): É um componente da molécula do ribossomo.

RNA de transferência (tRNA): Molécula de RNA que tem função de transportar o aminoácido até o ribossomo, onde ocorre a síntese da proteína.

Small RNA (sRNA): Classe das moléculas de RNA que possuem tamanho pequeno e que não codificam proteína.

Small nucleolar RNA snoRNA: Pequena molécula de RNA que realiza modificações químicas em outros RNAs.

Estresse abiótico: Danos na planta ocasionados por fatores não vivos, como clima e solo inadequados, excesso ou falta de água.

Estresse biótico: Danos na planta ocasionados por outros organismos vivos, como fungos e bactérias.

Volta (loop): A dobra do grampo, onde os nucleotídeos não realizam pareamento.

APÊNDICE A – PRE-miRNAS PREDITOS E SEUS PROVÁVEIS ALVOS

A Tabela A.1 apresenta os 160 pre-miRNAs preditos pelo classificador desenvolvido neste trabalho, juntamente com seus prováveis alvos.

Tabela A.1: Pre-miRNAs preditos e seus prováveis alvos.

Identificador	**Região**	**Alvo(s)**
Glyma0041s00200.1(1)	Intrônica	Glyma13g27410.1
Glyma01g09430.1(1)	Intrônica	Glyma02g23640.1, Glyma05g21210.1, Glyma20g20840.1.
Glyma01g22490.1(1)	Intrônica	Glyma14g33280.1.
Glyma01g28310.1(1)	Intrônica	Glyma06g03190.3, Glyma17g08030.1.
Glyma01g28880.1(1)	Intrônica	Glyma10g01560.1.
Glyma01g37140.1(1)	Intrônica	Glyma08g37270.1.
Glyma02g01820.1(1)	Intrônica	Glyma10g01890.1.
Glyma02g03610.1(1)	Intrônica	Glyma15g08560.1.
Glyma02g07110.1(1)	Intrônica	Glyma18g15920.1.
Glyma02g09970.1(1)	Intrônica	Glyma06g41100.1, Glyma06g41050.1, Glyma14g38810.1, Glyma02g40500.1.
Glyma02g15950.1(1)	Intrônica	Glyma09g24270.1, Glyma10g12900.1, Glyma07g00350.1.
Glyma02g16370.1(1)	Intrônica	Glyma17g02540.1, Glyma07g38180.1, Glyma17g02540.2.
Glyma02g23640.1(1)	Intrônica	Glyma02g23640.1, Glyma01g20840.1, Glyma14g32230.1, Glyma05g21210.1, Glyma01g09430.1, Glyma20g20840.1, Glyma09g23710.1, Glyma02g33010.1, Glyma07g34540.2, Glyma15g33010.1, Glyma10g22610.1.
		Glyma14g32230.1, Glyma05g21210.1, Glyma02g23640.1,

Tabela A.1 – Continuação.

Identificador	Região	Alvo(s)
Glyma02g23640.1(2)	Intrônica	Glyma01g20840.1, Glyma09g23710.1, Glyma02g33010.1, Glyma07g34540.2, Glyma20g20840.1, Glyma10g22610.1.
Glyma02g31250.1(1)	Intrônica	Glyma02g31250.1.
Glyma02g37020.1(1)	Intrônica	Glyma02g37020.1, Glyma17g07740.1, Glyma04g26780.2, Glyma04g26780.1.
Glyma02g45370.1(1)	Intrônica	Glyma17g00990.1.
Glyma03g00770.1(1)	Intrônica	Glyma15g03630.1, Glyma17g05970.1, Glyma13g16710.1.
Glyma03g27110.1(1)	Intrônica	Glyma10g30570.1, Glyma02g31470.1, Glyma05g10590.1.
Glyma03g28310.1(1)	Intrônica	Glyma02g41050.1.
Glyma03g34680.2(1)	Intrônica	Glyma03g34680.1.
Glyma03g35090.1(1)	Intrônica	Glyma11g05760.1.
Glyma04g06290.1(1)	Intrônica	Glyma13g01540.1.
Glyma04g07720.1(1)	Intrônica	Glyma15g04000.1.
Glyma04g16710.1(1)	Intrônica	Glyma09g12570.1, Glyma10g38600.1, Glyma04g11310.1, Glyma04g11310.2, Glyma04g11310.3, Glyma15g24130.1.
Glyma04g33990.1(2)	Intrônica	Glyma04g29490.1, Glyma10g25420.1, Glyma09g05950.1, Glyma12g18240.1.
Glyma04g37590.1(1)	Intrônica	Glyma06g17480.1, Glyma09g37900.1, Glyma13g33040.1.
Glyma04g43640.1(1)	Intrônica	Glyma10g06330.1.
Glyma05g03700.1(1)	Intrônica	Glyma07g28520.1, Glyma07g28520.2, Glyma17g14210.1, Glyma19g36230.1.
Glyma05g04290.1(1)	Intrônica	Glyma05g04290.2, Glyma14g35330.1, Glyma09g10500.1.
Glyma05g07420.1(1)	Intrônica	Glyma02g12830.1, Glyma17g08910.1, Glyma09g02060.1, Glyma08g41220.3, Glyma11g20470.1, Glyma01g36010.1, Glyma11g20470.3, Glyma11g20470.2, Glyma08g22860.1, Glyma09g34410.1, Glyma14g09670.1, Glyma11g02960.1, Glyma09g39350.1.
Glyma05g08200.1(1)	Intrônica	Glyma11g36910.1, Glyma08g42710.5, Glyma08g42710.1, Glyma06g43620.1, Glyma08g42710.3, Glyma08g42710.4, Glyma08g42710.2.

Tabela A.1 – Continuação.

Identificador	Região	Alvo(s)
Glyma05g21110.1(1)	Intrônica	Glyma07g02210.4.
Glyma05g21210.1(1)	Intrônica	Glyma02g23640.1, Glyma01g20840.1, Glyma05g21210.1, Glyma20g20840.1, Glyma14g32230.1, Glyma09g23710.1, Glyma02g33010.1, Glyma10g22610.1, Glyma07g34540.2, Glyma01g09430.1, Glyma03g13310.1, Glyma15g33010.1.
Glyma05g23150.1(1)	Intrônica	Glyma10g42690.1, Glyma02g11610.1, Glyma08g13640.1.
Glyma05g33000.1(1)	Intrônica	Glyma02g40460.1.
Glyma05g35680.1(1)	Intrônica	Glyma06g45310.1.
Glyma06g01140.1(1)	Intrônica	Glyma14g07590.2.
Glyma06g04870.1(1)	Intrônica	Glyma04g04790.1.
Glyma06g06340.1(1)	Intrônica	Glyma06g06340.1.
Glyma06g06440.1(1)	Intrônica	Glyma16g23090.1, Glyma04g41410.2, Glyma02g40170.1, Glyma19g30230.1, Glyma09g35670.1, Glyma04g37260.1, Glyma17g08200.1, Glyma13g09530.2, Glyma09g09800.1, Glyma02g36480.1.
Glyma06g09000.1(1)	Intrônica	Glyma15g08180.1.
Glyma06g40380.1(1)	Intrônica	Glyma06g40370.1, Glyma13g22990.1, Glyma06g40000.1, Glyma10g08350.1.
Glyma06g40820.1(1)	Intrônica	Glyma06g40710.1.
Glyma06g42460.1(1)	Intrônica	Glyma13g31390.1, Glyma10g04770.1, Glyma10g04770.2.
Glyma06g46320.1(1)	Intrônica	Glyma09g12380.1.
Glyma06g46640.1(1)	Intrônica	Glyma06g46640.2.
Glyma07g12010.1(1)	Intrônica	Glyma13g16800.1, Glyma02g16090.2, Glyma02g16090.1.
Glyma07g15650.1(1)	Intrônica	Glyma16g00450.2, Glyma16g00450.1, Glyma11g33150.1, Glyma12g06080.1, Glyma18g05080.1.
Glyma07g17010.1(1)	Intrônica	Glyma04g10880.1.
Glyma07g32720.1(1)	Intrônica	Glyma17g10920.1, Glyma17g10920.2, Glyma06g29610.1.
Glyma07g34620.1(1)	Intrônica	Glyma20g02370.2, Glyma20g02370.1.
Glyma08g05580.3(1)	Intrônica	Glyma08g05580.1, Glyma08g05580.2, Glyma05g34110.1, Glyma06g40130.1.

Tabela A.1 – Continuação.

Identificador	Região	Alvo(s)
Glyma08g06000.1(1)	Intrônica	Glyma07g07950.1.
Glyma08g09210.1(1)	Intrônica	Glyma10g02370.1, Glyma10g02370.2, Glyma13g24370.1, Glyma14g20110.1, Glyma12g03570.1.
Glyma08g19070.1(1)	Intrônica	Glyma17g00490.1.
Glyma08g19500.1(1)	Intrônica	Glyma19g42990.1, Glyma03g40440.2, Glyma03g40440.1, Glyma03g40440.3, Glyma03g40440.4, Glyma18g20960.1, Glyma08g38800.1, Glyma19g43070.1.
Glyma08g33510.1(1)	Intrônica	Glyma03g34570.1.
Glyma08g40690.1(1)	Intrônica	Glyma01g39280.1, Glyma08g04340.1.
Glyma08g42540.1(1)	Intrônica	Glyma03g25630.1.
Glyma08g42910.2(1)	Intrônica	Glyma08g42910.1, Glyma09g18560.1.
Glyma09g03000.1(1)	Intrônica	Glyma11g35910.1, Glyma01g33480.1.
Glyma09g04340.1(1)	Intrônica	Glyma13g24270.1, Glyma07g37070.1, Glyma10g08040.1, Glyma09g11460.1, Glyma03g38620.1, Glyma11g13720.2, Glyma02g14690.1, Glyma06g19830.2, Glyma13g37010.2, Glyma02g14570.1, Glyma13g37010.3.
Glyma09g23000.1(1)	Intrônica	Glyma19g22150.1, Glyma09g14060.1, Glyma09g19120.1, Glyma08g34290.1, Glyma0021s00660.1, Glyma01g19400.1, Glyma02g33820.1, Glyma06g32270.1, Glyma06g35310.1, Glyma11g28130.1, Glyma01g18300.1, Glyma06g30100.1, Glyma03g21340.1, Glyma02g30610.1, Glyma02g22320.1, Glyma01g17340.1, Glyma06g26150.1, Glyma09g23000.1, Glyma05g12900.1, Glyma11g22730.1, Glyma0023s00830.1, Glyma11g26530.1, Glyma18g29180.1, Glyma12g18190.1, Glyma11g17500.1, Glyma04g25380.1, Glyma13g08690.1, Glyma15g33290.1, Glyma14g20080.1, Glyma12g20440.1, Glyma15g25890.1, Glyma06g40190.1, Glyma17g23620.1, Glyma07g13520.1, Glyma17g27830.1, Glyma10g21800.1, Glyma07g21920.1, Glyma15g35670.1, Glyma02g19640.1, Glyma04g25390.1, Glyma10g21360.1, Glyma08g37260.1,

Tabela A.1 – Continuação.

Identificador	Região	Alvo(s)
		Glyma19g16980.1, Glyma11g26540.1, Glyma01g19410.1, Glyma02g29050.1, Glyma09g13590.1.
Glyma09g28920.1(1)	Intrônica	Glyma01g37610.2.
Glyma09g36480.1(1)	Intrônica	Glyma20g16230.2, Glyma13g10490.1, Glyma20g16230.1, Glyma13g10490.2, Glyma20g16230.3, Glyma11g15590.1, Glyma18g49000.1.
Glyma09g36940.1(1)	Intrônica	Glyma08g39480.1, Glyma17g17050.2, Glyma12g05010.2, Glyma17g17050.4, Glyma17g17050.1, Glyma17g17050.3, Glyma12g05010.1.
Glyma09g38130.1(1)	Intrônica	Glyma08g20740.1.
Glyma10g03470.1(1)	Intrônica	Glyma06g10210.1.
Glyma10g04290.1(1)	Intrônica	Glyma13g18460.1.
Glyma10g07310.1(1)	Intrônica	Glyma06g12370.1, Glyma11g05500.1, Glyma01g39790.1.
Glyma10g12660.1(1)	Intrônica	Glyma10g12660.1.
Glyma10g31230.1(1)	Intrônica	Glyma10g01030.1, Glyma10g01030.2.
Glyma10g34530.2(1)	Intrônica	Glyma10g34530.1, Glyma10g20550.1, Glyma12g08500.1, Glyma12g01720.1, Glyma15g08000.1, Glyma08g44650.1, Glyma16g13580.1, Glyma06g18430.1, Glyma16g27550.1, Glyma08g18070.1, Glyma12g28860.1, Glyma15g42500.1, Glyma07g16230.1, Glyma17g05840.1, Glyma17g05840.2, Glyma17g37110.1, Glyma02g00260.1, Glyma20g26300.2, Glyma03g39780.1, Glyma02g04750.1, Glyma18g43530.1, Glyma16g26170.1, Glyma03g04370.1, Glyma02g01870.1.
Glyma10g35640.1(1)	Intrônica	Glyma20g31910.1, Glyma16g26100.2, Glyma16g26100.1, Glyma02g07110.1, Glyma01g01390.1, Glyma10g42330.1.
Glyma10g36790.1(1)	Intrônica	Glyma09g40740.2, Glyma09g40740.1.
Glyma10g38330.1(1)	Intrônica	Glyma10g13860.1, Glyma02g18890.2, Glyma02g18890.1.
Glyma10g41000.1(1)	Intrônica	Glyma14g01540.1.
Glyma10g42830.1(1)	Intrônica	Glyma16g03780.1.
Glyma11g02920.1(1)	Intrônica	Glyma02g10880.1.

Tabela A.1 – Continuação.

Identificador	Região	Alvo(s)
Glyma11g06270.1(1)	Intrônica	Glyma05g07880.1, Glyma19g00820.1.
Glyma11g06900.1(1)	Intrônica	Glyma03g31080.1.
Glyma11g11840.1(1)	Intrônica	Glyma09g34830.1.
Glyma11g13770.1(1)	Intrônica	Glyma09g03480.1, Glyma09g03480.2.
Glyma11g17120.1(1)	Intrônica	Glyma02g10530.1, Glyma18g52350.1, Glyma15g22050.1, Glyma10g43700.1.
Glyma11g18300.1(1)	Intrônica	Glyma07g15620.1, Glyma01g00520.1.
Glyma12g00770.1(1)	Intrônica	Glyma01g01570.1.
Glyma12g05790.1(1)	Intrônica	Glyma09g34320.1, Glyma10g42330.1, Glyma07g37840.1, Glyma01g43650.1.
Glyma12g05820.1(1)	Intrônica	Glyma01g27040.1.
Glyma12g06130.1(1)	Intrônica	Glyma19g33860.1, Glyma19g33860.2, Glyma10g39300.1.
Glyma12g09980.1(1)	Intrônica	Glyma01g02640.2, Glyma01g02640.1.
Glyma12g11070.1(1)	Intrônica	Glyma10g43430.1.
Glyma12g13560.1(1)	Intrônica	Glyma15g23610.1, Glyma04g00550.2, Glyma18g49630.1, Glyma04g00550.1.
Glyma12g28950.1(1)	Intrônica	Glyma16g03490.1.
Glyma12g29960.1(2)	Intrônica	Glyma17g04690.1.
Glyma12g32450.1(1)	Intrônica	Glyma12g32460.1, Glyma07g09510.1.
Glyma12g35120.1(1)	Intrônica	Glyma18g47630.1, Glyma08g18200.2.
Glyma12g36100.1(1)	Intrônica	Glyma12g36100.1.
Glyma13g09370.1(1)	Intrônica	Glyma11g11750.2, Glyma11g11750.1, Glyma10g27710.1.
Glyma13g10410.1(1)	Intrônica	Glyma08g03680.1.
Glyma13g10560.1(1)	Intrônica	Glyma16g33080.1, Glyma09g40590.1, Glyma09g28260.1.
Glyma13g17710.1(1)	Intrônica	Glyma07g21100.1, Glyma17g05520.1, Glyma03g41420.2, Glyma18g00500.1, Glyma17g29170.2, Glyma11g08540.1, Glyma18g20960.1, Glyma02g40780.1, Glyma08g19140.1, Glyma13g17490.2, Glyma13g17490.1, Glyma02g03740.1, Glyma17g02020.1.
Glyma13g20850.1(1)	Intrônica	Glyma10g06640.1, Glyma10g31960.1, Glyma11g34580.1.

Tabela A.1 – Continuação.

Identificador	Região	Alvo(s)
Glyma13g23350.1(1)	Intrônica	Glyma17g37720.1.
Glyma13g27300.1(1)	Intrônica	Glyma02g44580.1.
Glyma13g38750.1(1)	Intrônica	Glyma15g20970.1, Glyma03g18410.2.
Glyma13g39600.1(1)	Intrônica	Glyma14g36400.1, Glyma11g12070.1, Glyma04g10590.1, Glyma11g12070.2.
Glyma13g39930.1(1)	Intrônica	Glyma09g31920.2, Glyma05g26810.1, Glyma06g27440.2, Glyma02g15370.2, Glyma04g36430.1, Glyma18g06930.1, Glyma07g06070.1.
Glyma14g00940.1(1)	Intrônica	Glyma13g43130.1, Glyma13g43130.2.
Glyma14g22180.1(1)	Intrônica	Glyma04g39470.1.
Glyma14g24590.1(1)	Intrônica	Glyma08g45150.2, Glyma13g03740.1, Glyma08g45150.1, Glyma08g45150.3.
Glyma14g28740.1(1)	Intrônica	Glyma11g02210.1, Glyma11g04150.1, Glyma04g08930.1.
Glyma14g36580.1(1)	Intrônica	Glyma11g12920.2, Glyma11g12920.1, Glyma09g04890.1.
Glyma14g39420.1(1)	Intrônica	Glyma16g03310.1.
Glyma14g39450.1(1)	Intrônica	Glyma10g14700.1.
Glyma14g39790.1(1)	Intrônica	Glyma18g06360.1.
Glyma15g01550.1(1)	Intrônica	Glyma18g01750.1.
Glyma15g01550.2(1)	Intrônica	Glyma18g01750.1.
Glyma15g03620.1(1)	Intrônica	Glyma13g40430.1, Glyma16g23160.1.
Glyma15g04200.1(1)	Intrônica	Glyma10g33900.1.
Glyma15g05090.1(1)	Intrônica	Glyma02g15680.1.
Glyma15g05470.1(1)	Intrônica	Glyma14g05630.1.
Glyma15g18580.1(1)	Intrônica	Glyma17g13970.1, Glyma19g36740.1, Glyma16g34070.1, Glyma04g41460.1.
Glyma15g35960.1(1)	Intrônica	Glyma02g38050.1.
Glyma15g37830.1(1)	Intrônica	Glyma03g28870.1, Glyma13g31060.1.
Glyma15g40430.1(1)	Intrônica	Glyma15g05970.1.
Glyma16g02290.1(1)	Intrônica	Glyma07g01130.1.
Glyma16g17450.1(1)	Intrônica	Glyma15g23610.1, Glyma04g00550.2, Glyma04g00550.1.

Tabela A.1 – Continuação.

Identificador	Região	Alvo(s)
Glyma16g25110.1(1)	Intrônica	Glyma13g17980.1, Glyma12g05720.1, Glyma20g11740.1.
Glyma16g29760.1(1)	Intrônica	Glyma09g24210.1.
Glyma15g33290.1(1)	Intrônica	Glyma11g26530.1, Glyma06g32270.1, Glyma11g22730.1, Glyma05g12900.1, Glyma11g17500.1, Glyma09g23000.1, Glyma12g18190.1, Glyma0021s00660.1, Glyma09g14060.1, Glyma11g28130.1, Glyma15g33290.1, Glyma01g18300.1, Glyma06g30100.1, Glyma09g19120.1, Glyma02g22320.1, Glyma08g34290.1, Glyma02g30610.1, Glyma01g17340.1, Glyma06g35310.1, Glyma19g22150.1, Glyma13g08690.1, Glyma01g19400.1, Glyma03g21340.1, Glyma06g26150.1, Glyma17g23620.1, Glyma0023s00830.1, Glyma04g25380.1, Glyma02g33820.1, Glyma18g29180.1, Glyma12g20440.1, Glyma15g25890.1, Glyma17g27830.1, Glyma14g20080.1, Glyma06g40190.1, Glyma07g13520.1, Glyma02g19640.1, Glyma10g21360.1, Glyma07g21920.1, Glyma15g35670.1, Glyma10g21800.1, Glyma19g16980.1, Glyma11g26540.1, Glyma04g25390.1, Glyma01g19410.1, Glyma08g37260.1, Glyma09g13590.1, Glyma02g29050.1.
Glyma16g18390.1(1)	Intrônica	Glyma06g19630.1, Glyma06g19640.1, Glyma15g41170.1, Glyma17g15840.2, Glyma10g00660.1, Glyma09g37020.2, Glyma01g00400.2, Glyma04g35110.2, Glyma04g35110.1, Glyma20g37770.1, Glyma01g00400.1, Glyma04g35100.2, Glyma01g00400.3, Glyma01g00400.4.
Glyma16g32730.1(1)	Intrônica	Glyma16g32710.1, Glyma16g32680.1, Glyma11g18710.1.
Glyma17g01940.1(1)	Intrônica	Glyma02g13980.1, Glyma07g21120.1, Glyma05g31450.1, Glyma05g31450.2, Glyma05g31450.3, Glyma08g10000.1.
Glyma17g07250.1(1)	Intrônica	Glyma17g21570.1, Glyma15g06900.1.
Glyma17g18890.1(1)	Intrônica	Glyma17g17880.1, Glyma20g04820.1.
Glyma17g22650.1(1)	Intrônica	Glyma13g28060.1, Glyma03g39760.1, Glyma06g47160.1.
Glyma17g31360.1(1)	Intrônica	Glyma19g41590.1, Glyma13g03650.1.

Tabela A.1 – Continuação.

Identificador	Região	Alvo(s)
Glyma17g37910.1(1)	Intrônica	Glyma04g42370.1, Glyma06g12430.1.
Glyma17g38040.1(1)	Intrônica	Glyma07g13900.1.
Glyma18g01950.1(2)	Intrônica	Glyma19g22460.1.
Glyma18g05800.1(1)	Intrônica	Glyma07g04260.1.
Glyma18g07700.1(1)	Intrônica	Glyma09g26710.1.
Glyma18g19720.1(1)	Intrônica	Glyma18g46970.1.
Glyma18g33480.1(1)	Intrônica	Glyma02g23640.1, Glyma01g20840.1, Glyma01g09430.1, Glyma05g21210.1, Glyma14g32230.1, Glyma20g20840.1, Glyma09g23710.1, Glyma02g33010.1, Glyma10g22610.1, Glyma07g34540.2, Glyma15g33010.1.
Glyma18g33910.1(1)	Intrônica	Glyma15g23610.1, Glyma04g00550.2, Glyma04g00550.1.
Glyma18g48520.1(1)	Intrônica	Glyma07g35560.1, Glyma0041s00360.1.
Glyma18g53420.1(1)	Intrônica	Glyma19g02510.1.
Glyma18g54010.1(1)	Intrônica	Glyma11g25860.1, Glyma13g26110.1.
Glyma19g02330.1(1)	Intrônica	Glyma13g45050.1, Glyma19g02340.1, Glyma15g00280.1.
Glyma19g02340.1(1)	Intrônica	Glyma13g45050.1.
Glyma19g02380.1(1)	Intrônica	Glyma03g30650.1, Glyma09g03850.1, Glyma08g42870.1.
Glyma19g02890.1(1)	Intrônica	Glyma12g35600.1, Glyma11g05590.1, Glyma01g38650.1, Glyma02g40780.1, Glyma01g38650.2.
Glyma19g24870.1(1)	Intrônica	Glyma15g18700.1, Glyma11g06780.1, Glyma15g23100.1.
Glyma19g27360.1(1)	Intrônica	Glyma16g34070.1.
Glyma19g29750.1(1)	Intrônica	Glyma07g08290.1, Glyma08g47100.1, Glyma06g32870.1, Glyma12g18240.1.
Glyma20g02370.1(1)	Intrônica	Glyma04g32120.1.
Glyma20g14280.1(1)	Intrônica	Glyma08g28040.2, Glyma18g08530.1, Glyma09g34420.1, Glyma04g42380.1.
Glyma20g25140.1(1)	Intrônica	Glyma14g07210.1, Glyma14g07210.3, Glyma08g43880.2, Glyma08g43880.1, Glyma06g20900.1, Glyma05g27060.1, Glyma08g43880.3, Glyma14g07210.2.
Glyma20g26980.1(1)	Intrônica	Glyma14g29100.1, Glyma08g01580.1, Glyma05g38000.1.

APÊNDICE B – ARTIGO PUBLICADO COM TEMA RELACIONADO À DISSERTAÇÃO

Segue artigo com tema relacionado ao desta dissertação, publicado nos anais do *10th International Conference on Intelligent Systems Design and Applications (ISDA'10)* (Qualis B3) que aconteceu na cidade do Cairo, Egito de 29/11 a 01/12/2010. Este artigo está disponível na biblioteca digital do IEEE (IEEE-Xplore^{TM}).

Using a Support Vector Machine to identify pre-miRNAs in soybean (*Glycine max*) introns

Paulo R. Silla; Maria Angélica de O. Camargo-Brunetto
Computer Science Department
State University of Londrina, UEL
Londrina-PR, Brazil
silla@cnpso.embrapa.br; angelica@uel.br

Eliseu Binneck
Laboratory for Bioinformatics
Brazilian Agricultural Research Corporation,
Embrapa Soybean
Londrina-PR, Brazil
binneck@cnpso.embrapa.br

***Abstract*—MicroRNAs (miRNAs) are small Ribonucleic Acid (RNA) molecules ∼18–22 nucleotides (nt) in length that regulates gene expression in animals, plants and viruses. Due to its small size and occurrence in different development stages of organisms, the experimental identification of miRNAs becomes difficult, and computational approaches are being developed in order to precede and guide biological experiments. This paper describes our approach based on a Support Vector Machine (SVM) algorithm to identify miRNA's precursor (pre-miRNA) in soybean (*Glycine max*) transcript introns, that was developed using a secondary structure predictor of pre-miRNAs sequences to establish the feature set for training, testing and validation phases of SVM algorithm.**

***Keywords*-Support Vector Machine; pre-miRNA; soybean (*Glycine max*)**

I. Introduction

Soybean is one of the most important agricultural crops in the world due to its role in the food industry and biofuel production. Brazil is the second largest producer, with an estimated production of 67.86 million tons for the 2009/2010 season using an area of 23.24 million ha [1]. Due to the numerous challenges in agriculture, there is a clear need to develop better adapted and more productive soybean cultivars.

One of the recent advances in biological research was the discovery of small RNA molecules, called microRNA (miRNA) that can regulate gene expression at post-transcriptional level [2]. miRNAs are known to play important roles in soybean development, nitrogen fixation, and the response to abiotic and biotic stresses [3].

Computational approaches for discovering miRNAs can be performed as follows: i) homology-based search [4] – the comparison of candidate sequences against known miRNAs, however this technique does not identify new miRNAs; ii) *ab initio* prediction – uses attributes of the precursor miRNA (pre-miRNA) sequence to establish new miRNAs.

The discovery of miRNA is still a challenge and many studies are in progress. Methods based on machine-learning have recently been successfully applied to miRNA discovery [5], [6] [7]. These methods require a set of known pre-miRNAs and random sequences to be used as training sets. After the training phase, tests should be conducted to measure how accurate are the predictions of the machine-learning algorithm. Finally, candidate sequences are presented to the algorithm, which classifies each sequence into one of two classes: pre-miRNA or nonpre-miRNA.

Thus, the classifier indicates which candidate sequences are closer to known sequences and therefore determines those sequences likely to be experimentally verified as an actual pre-miRNA.

Instead of directly using miRNA sequences, machine-learning methods work with pre-miRNAs, since they fold into pre-miRNA hairpin and a set of sequence features can be determined (see IV-B) to distinguish pre-miRNAs from other noncoding RNAs (ncRNAs).

The objectives of this work are: i) to develop a plant pre-miRNA sequence classifier using the SVM learning algorithm; ii) to identify feasible pre-miRNA sequences in soybean (*Glycine max*) transcript introns by using the classifier developed; iii) to verify the similarity of feasible pre-miRNAs identified by SVM with protein coding sequences from *Glycine max* genome.

This paper is organized as follows: Sec. II shows biological concepts for miRNAs; Sec. III presents previous works to the study conducted here; Sec. IV shows the used datasets (Sec. IV-A), the features calculated for each pre-miRNA sequence (Sec. IV-B), the developed scripts for the acquisition and pre-processing of data (Sec. IV-C), the SVM algorithm (Sec. IV-D), the SVM training, testing and validation phases and classification of candidate sequences (Sec. IV-E) and the pre-miRNA verification using a similarity searching algorithm (Sec. IV-F); Sec. V presents the SVM algorithm training results and also the pre-miRNAs identified by classifier and verified in a similarity search; Sec. VI exposes our conclusions and future works.

II. Biological concepts

According to Mendes *et al.* in [4], the miRNA precursor (pre-miRNA) of eukaryotes organisms originates in the cell's nucleus mainly in two ways: i) a primary miRNA (pri-miRNA) is processed by Drosha enzyme or, ii) from a

particular kind of intron (part of a gene's sequence that is not translated into protein), that appears after the splicing in precursor messenger RNA (pre-mRNA). Next, the pre-miRNA is exported to the cytoplasm (by enzymes carriers), where it undergoes the maturation process (by Dicer enzyme in animals or by Dicer like enzymes in plants). In this process, the loop present in hairpin is released, leaving only two strands that consist in a structure known as miRNA:miRNA*. Finally, the functional miRNA strand is integrated into miRNA-induced silencing complex (miRISC) that binds to a target mRNA and regulates the translation process in two possible ways: i) cleave the mRNA causing immediate degradation; ii) promote attenuation and therefore target degradation.

The study of [8] points out some features that can distinguish pre-miRNAs from mRNAs, transporter RNAs (tRNAs) and ribosomal RNAs (rRNAs): i) pre-miRNAs have low similarity, are less conserved, have higher AMFE and higher base pair rate than other RNAs; ii) pre-miRNA's MFEI_1 is greater than 0.85; iii) pre-miRNAs have low A+U content with respect to mRNAs and higher A+U content compared to other ncRNAs. AMFE, base pair rate and MFEI_1 will be described in Sec. IV-B.

III. Related Work

Zhang *et al.* in [3], performed a homology search in soybean (*Glycine max*) expressed sequence tags (ESTs), using 184 *Arabidopsis thaliana* miRNAs as reference and found 69 miRNAs.

Sewer *et al.* in [6], relied on the heuristic that animal miRNAs occur in clusters to establish candidate sequences and then used a SVM to make the prediction. From 3829 pre-miRNA candidate sequences in humans, 3537 in mice and 3034 in rats, 224, 192 and 208 were classified as human, mouse and rat pre-miRNA respectively. After removing known sequences, remains 89 new pre-miRNAs in humans, 66 in mice and 105 in rats.

The miPred, developed by Loong and Mishra in [7], is a *de novo* SVM classifier model that uses kernel Radial Basis Function (RBF). The authors obtained 29 attributes of each sequence and trained the algorithm with 200 human pre-miRNA and 400 pseudo hairpins. miPred showed predictive performance equal or superior to other existing methods (ProMir [9], 3SVM [10], BayesMirFinder [11], RNAmicro [12] and [6]) regarding sensitivity and specificity, since it was able to identify pre-miRNA from different species of organisms with high accuracy, e.g. in a test set consisting of 1918 pre-miRNAs and 3836 pseudo hairpins, miPred presented sensitivity, specificity and accuracy equals to 87.65%, 97.75% and 94.38% respectively.

There is not yet, in related work, the application of a methodology based on machine learning in order to identify new pre-miRNA sequences in soybean genome. Another issue not well known is the presence of pre-miRNAs in plant introns. In this context, this work presents itself as an *ab initio* approach applied to find new soybean pre-miRNA sequences in soybean transcript introns.

Since miPred [7] presented itself as the best predictor, we based our work in producing the same 29 sequence features and, as we intend to investigate the existence of pre-miRNAs in soybean introns, the classifier training was conducted with the use of known plant pre-miRNAs (positive set) plus other types of ncRNA and random sequences (negative set).

IV. Materials and methods

This section describes the methodology used to develop this work, which also can be seen in Fig. 1.

A. Datasets

The positive set was created with a total of 430 known plant pre-miRNAs sequences, downloaded from the Plant MicroRNA Database[1] [13], of which 131 are from *Glycine max*, 199 are from *Arabidopsis thaliana* and 100 are from *Medicago truncatula*.

The negative data set was composed by 400 sequences, of which 175 are *Arabidopsis thaliana* small nucleolar RNA (snoRNA) sequences downloaded from the deepBase[2] [14] plus 225 RNA sequences that were generated randomly.

The pre-miRNA candidate sequences were obtained from existing introns in known *Glycine max* transcripts which were downloaded from Phytozome[3] [15], by a Biomart[4] tool and pre-processed as described in Sec. IV-C.

B. Set of features

Once the data sets have only nucleotide sequences and SVM requires a numeric input, it is necessary to encode this sequences in order to identify the pre-miRNAs. Then, each sequence s will be represented by 29 RNA global and intrinsic folding attributes [7] [16]. Setting $L(s)$ as the s length, the features were achieved as following: **dinucleotide[5] frequency** – $\%XY$ such that $X, Y \in \sum = [A, C, G, U]$, computed by:

$$\%XY(s) = F_{XY}(s)/L(s) \times 100, \tag{1}$$

where $F_{XY}(s)$ denotes the absolute frequency of dinucleotide XY in s.

Aggregate dinucleotide frequency – $\%(G + C)$ ratio, computed by:

$$Ratio_{GC}(s) = (F_G(s) + F_C(s))/L(s) \times 100, \tag{2}$$

where $F_G(s)$ and $F_C(s)$ represent the absolute frequency of G and C in s respectively.

[1] http://bioinformatics.cau.edu.cn/PMRD/
[2] http://deepbase.sysu.edu.cn/
[3] www.pythozome.net
[4] www.phytozome.net/biomart/martview
[5] The nucleotides present in RNA are: Adenine (A), Cytosine (C), Guanine (G) and Uracil (U).

Adjusted minimum free energy of folding (AMFE) [8] – represents the Minimal Folding Free Energy (MFE) in kcal/mol of a 100 nt sequence, computed by:

$$AMFE(s) = MFE(s)/L(s) \times 100, \quad (3)$$

where MFE was computed using the RNAlib[6] library (Vienna RNA Package) [17] an ANSI C implementation of Zuker's algorithm, that performs a complete evaluation of all feasible structures for a given RNA sequence, and determine the one with minimal free energy [18].

MFE Index 1 (MFEI$_1$) [8] – ratio of $AMFE(s)$ and $Ratio_{GC}(s)$ calculated by:

$$MFEI_1(s) = AMFE(s)/Ratio_{GC}(s). \quad (4)$$

MFE Index 2 (MFEI$_2$) [16] – ratio of $MFE(s)$ and number of stems, that are structural motifs formed by more than two base pairs in sequence, computed by:

$$MFEI_2(s) = AMFE(s)/stem(db_s), \quad (5)$$

where $stem()$ function receives s secondary structure in dot-bracket notation db_s as input, obtained by RNAlib library, and returns the total number of stems.

Adjusted base pairing propensity (P) [19] – the total number of base pairs normalized by $L(s)$, obtained by:

$$P(s) = BP(s)/L(s), \quad (6)$$

where $BP(s)$, obtained by RNAlib, represents the base pair number in s secondary structure.

Adjusted base pair distance (D) [20] and **normalised Shannon's entropy (Q)** [21] – calculated by (7) and (8) respectively, where ρ_{ij} matrix is obtained by RNAlib through McCaskill's algorithm and denotes the probability for base s_i and s_j to form a pair.

$$D(s) = \sum_{i<j} (\rho_{ij} - \rho_{ij}^2)/L(s) \quad (7)$$

$$Q(s) = -\sum_{i<j} \rho_{ij} log_2(\rho_{ij})/L(s) \quad (8)$$

Degree of compactness (F) (second eigenvalue), which characterizes the complexity of RNA topology [22] – was calculated by an implementation of RNAspectral algorithm [23]. The RNAspectral takes db_s as input, obtained by RNAlib, it creates the representation of RNA's secondary structure as a planar tree-graph and then calculates its Laplacian matrix, so that the Laplacian matrix is the difference between graph matrices degree and adjacency. The degree of compactness is the second smallest nonzero value from the eigenvalue set of graph's Laplacian matrix.

The last 5 features are based on [16] who argues that z-score of AMFE, P, D, Q and F features are statistically able to distinguish pre-miRNA sequences from other types of ncRNAs and mRNAs, even considering the small amount of data (1806 know pre-miRNAs, 12,387 ncRNAs and 31 mRNAs). The z-scores of these attributes are calculated by:

$$ZScore(A) = (A_s - \mu(A_{shuffled}))/\sigma(A_{shuffled}), \quad (9)$$

where A_s represents the value of feature A of sequence s, $A_{shuffled}$ is a vector of feature A values, calculated for each of the 10^4 sequences obtained by nucleotide shuffling in s, $\mu()$ function returns the arithmetic mean and $\sigma()$ function returns the Standard Deviation. In this way, the 29 features where completed.

C. Pre-processing

The set of features must be computed for all datasets, including the pre-miRNA candidates. Before describing the script developed for this task, it will be explained the building of the pre-miRNA candidate sequences set.

To obtain this set, we created two Python[7] scripts, as follows: Script 1) which receives as input the soybean transcripts in a fasta[8] format file and, as output, writes only the introns of each transcript in another fasta file; Script 2) that receives as input the output file from Script 1 and performs two filters, in order to exclude meaningless sequences: $44 \leq L(s) \leq 259$ nt [3] and MFEI$_1 > 0.85$ [8]. The resultant sequences from Script 2 are also recorded in fasta-formatted file.

After the use of Python scripts 1 and 2, we obtained a total of $1,205$ pre-miRNA candidate sequences.

The 29 attributes (see Section IV-B) were computed for all sequences, by a parallel Python script (Script 3), that receives as input a fasta-formatted file containing positive, negative and candidate sequences and stores its results in a MySQL[9] database, called sequence attributes database (see Fig. 1a).

In sequence attributes database, sequences are separated by expected value, as follows: sequences with expected value equals to "1" represent the positive set; sequences with expected value equals to "-1" represent the negative set; pre-miRNA candidate sequences have an expected value equals to "0". Thus, the sequences and values of its attributes are available for future usage.

Swig[10] tool was used to establish an interface between Scripts and RNAlib. Fig. 1b shows the Script 3 in details.

This script was implemented using the Parallel Python[11] module as described below: a process was created for each sequence to be analysed. These processes were placed in a queue and then allocated to one available core in the parallel machine, which was composed of: i) Sun Fire X4150, 2 Intel Quad-Core (R) processors, 32 Gb RAM; ii) Sun Fire X4450, 2 Intel Six-Core (R) processors, 32 Gb RAM.

[6] http://www.tbi.univie.ac.at/~ivo/RNA/

[7] www.python.org

[8] File format used in bioinformatics for recording sequences.

[9] www.mysql.com

[10] www.swig.org

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

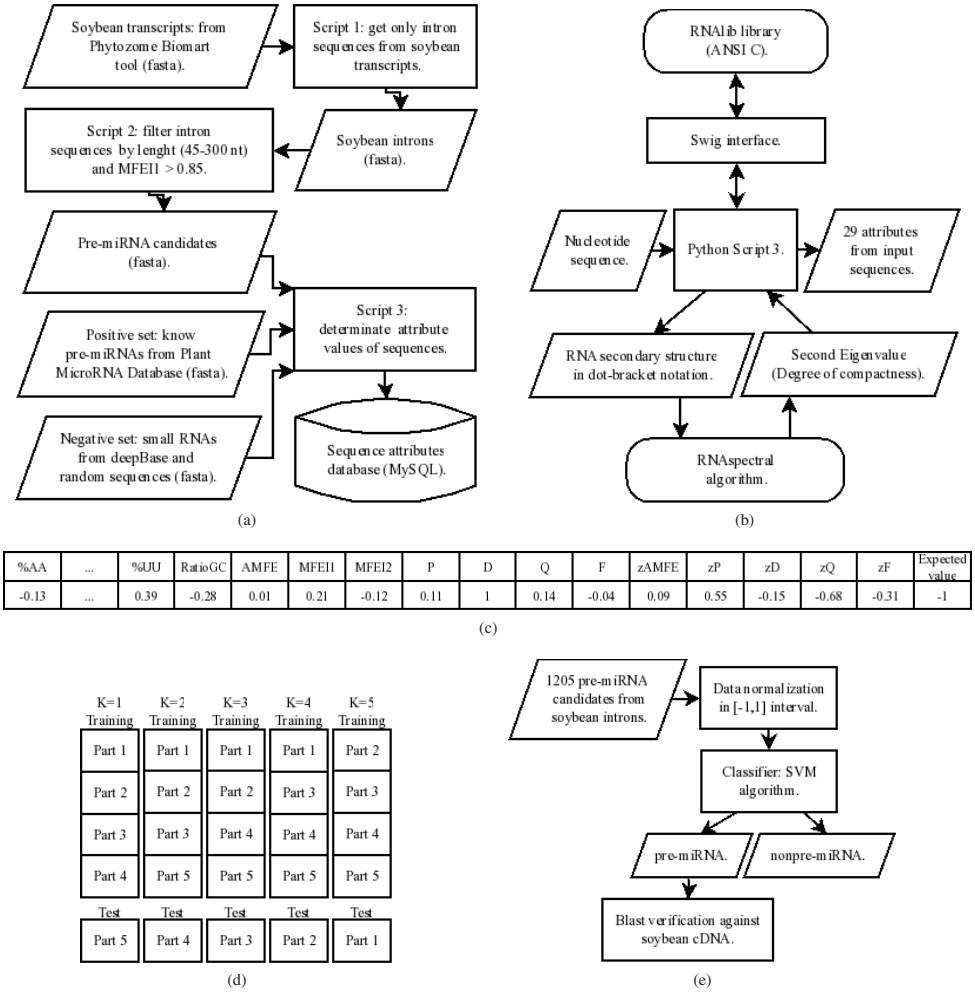

Figure 1. Methodology employed in this work: (a) sequences are obtained from different sources and, after obtaining the value of attributes, they are centralized in sequence attributes database; (b) detailed vision into getting sequence attribute values (Script 3); (c) example of an input vector to the SVM algorithm; (d) 5-fold method used in training/test of SVM algorithm to achieve the best (C, σ^2) pair. (e) SVM algorithm usage after training and validation phases, to classify the candidate sequences and the verification process through BLAST algorithm.

D. SVM Algorithm

SVM is a classification algorithm that provides state-of-the-art performance in a wide variety of application domains [24] and in recent years have been widely used to bioinformatics applications. Moreover, according to [25], SVM has an inherent ability to solve a pattern classification so close to the optimum without any incorporated knowledge of the problem domain into the machine's design.

We implemented the Sequential Minimal Optimization (SMO) [26] to conduct the training of SVM, since SMO algorithm solves the dual problem (see [25]) in a fast and efficient way. We used the gaussian RBF [25] (Eq. 10) as kernel function, because it is often applied giving good results [6] [7].

$$K(x, x_i) = exp(-(1/2\sigma^2)\|x - x_i\|^2), \qquad (10)$$

where $exp()$ is the exponential function, σ^2 is a kernel parameter that represents the common width to all kernels and x_i indicates the $i-th$ sample from x input data.

E. SVM training, testing and validation phases and pre-miRNA classification

The SVM training set (TS) was constructed with 400 sequences, as follows: 131 *Glycine max* pre-miRNA sequences and 69 *Arabidopsis thaliana* pre-miRNA from the positive set plus 175 *Arabidopsis thaliana* snoRNA and 25 random sequences from the negative set. The validation set (VS) was created using 200 sequences, as follows: 100 *Medicago truncatula* pre-miRNA from the positive set plus 100 random sequences from the negative set.

All data presented to the SVM algorithm (see Fig. 1c) were normalized by feature, in $[-1.0, 1.0]$ interval by:

$$\hat{X}_i = 2 \times ((X_i - X_{min})/(X_{max} - X_{min})) - 1, \quad (11)$$

where X contains the values that are going to be normalized, X_i represents the $i-th$ X value, X_{max} and X_{min} are highest and lowest X values respectively.

SVM training requires two parameters: constant regularization C from SMO algorithm and σ^2 from kernel function. The (C, σ^2) pair must be chosen in order to maximize the accuracy (acc), sensitivity (sens) and specificity (spec) of the classifier, which are computed by Eqs. (12–14), where TP, TN, FP and FN indicate the number of true positives, true negatives, false positives and false negatives results respectively.

$$acc = (TP + TN)/(TP + TN + FP + FN), \quad (12)$$

$$sens = (TP/(TP + FN)) \times 100, \quad (13)$$

$$spec = (TN/(TN + FP)) \times 100, \quad (14)$$

In accordance with the proposal of [27] we have tested 100 possibilities from a 10x10 grid as follows: $(C_i, \sigma_j^2) \mid C_i, \sigma_j^2 \in \{2^{-4}, 2^{-3}, ..., 2^4, 2^5\}$, and used the k-fold method, with k=5 (see Fig. 1d), to obtain the best values for (C, σ^2) pair.

In this way, TS was partitioned into 5 equal parts, 4 of them were used for training and 1 was used to test. This process was repeated until all subsets have been used for testing. The final result is the average of each k result. We developed a parallel program for the 5-fold, so that each training was performed by one process.

After obtaining the best (C, σ^2) pair, SVM was trained a last time using the TS set (without k-fold), subjected to a validation process by the VS set and used to classify the candidate sequences (Fig. 1e).

F. pre-miRNA candidates verification

Plant miRNAs typically bind their targets with near perfect complementarity [28]. Hence, the computational verification of sequences classified by SVM algorithm as pre-miRNA and therefore the prediction of their targets can be performed with a similarity search algorithm. In this step, we used the Basic Local Alignment Search Tool (BLAST[12]) [29].

We performed similarity search, using blastn algorithm (nucleotide x nucleotide), against *Glycine max* complementary DNA (cDNA) sequences (also downloaded from Phytozome [15]), because they only contain protein coding sequences.

Since candidate pre-miRNA sequences used had $45 - 300$ nt, we set the option word size (-W) to 18 in order that the hits have at least 18 nt. The Expectation Value (E-Value), that indicates the probability of an alignment occurring by chance, was assigned by default value (10).

V. RESULTS

The search grid resulted in $(C = 1.0, \sigma^2 = 1.0)$ as the best pair. Next, we conducted searches in the $(C = 1.0, \sigma^2 = 1.0)$ neighbourhood, which the best result was $(C = 1.5, \sigma^2 = 1.1)$. Finally, a validation step was performed in SVM algorithm by VS set. Table I shows accuracy, sensitivity and specificity results from training and validation steps performed in SVM algorithm.

Table I
SVM RESULTS FOR THE TRAINING AND VALIDATION PHASES.

Dataset	C	σ^2	acc.	sens.	spec.
TS (using k-fold).	1.0	1.0	0.92	91.34%	93.77%
TS (using k-fold).	1.5	1.1	0.93	91.30%	94.84%
VS (without k-fold).	1.5	1.1	0.92	89.0%	95.0%

After the training and validation phases, the SVM algorithm was used to classify the candidate sequences. In this step, from 1205 sequences, 329 were classified as pre-miRNA.

Since biological experiments to validate the pre-miRNAs classified by SVM are expensive and difficult to be made, we performed a verification by an *in silico* approach, using the BLAST algorithm.

This verification showed that 160 from 329 sequences classified as pre-miRNA have 18 or more nt similarity to at least one soybean protein coding sequence.

The results after BLAST validation can be seen in Table II. In this table, the column prefix indicates the transcript

[12]ftp://ftp.ncbi.nih.gov/blast/

location within the genome, e.g. "Glyma03g" prefix means that the transcripts at this line are present in *Glycine max* chromosome 3, the column Transcript ids (intron) contains the remaining portion of the transcript name and shows the intron number in parentheses. Introns were numbered in the order of appearance in the transcript.

Table II
FEASIBLE PRE-MIRNA'S AFTER BLAST VERIFICATION

Prefix	Transcript ids (intron)
Glyma0041s	00200.1 (1).
Glyma01g	09430.1 (1), 22490.1 (1), 28310.1 (1), 28880.1 (1), 37140.1 (1).
Glyma02g	01820.1 (1), 03610.1 (1), 07110.1 (1), 09970.1 (1), 15950.1 (1), 16370.1 (1), 23640.1 (1), 23640.1 (2), 31250.1 (1), 37020.1 (1), 45370.1 (1).
Glyma03g	00770.1 (1), 27110.1 (1), 28310.1 (1), 34680.2 (1), 35090.1 (1).
Glyma04g	06290.1 (1), 07720.1 (1), 16710.1 (1), 33990.1 (2), 37590.1 (1), 43640.1 (1).
Glyma05g	03700.1 (1), 04290.1 (1), 07420.1 (1), 08200.1 (1), 21110.1 (1), 21210.1 (1), 23150.1 (1), 33000.1 (1), 35680.1 (1).
Glyma06g	01140.1 (1), 04870.1 (1), 06340.1 (1), 06440.1 (1), 09000.1 (1), 40380.1 (1), 40820.1 (1), 42460.1(1), 46320.1 (1), 46640.1 (1).
Glyma07g	12010.1 (1), 15650.1 (1), 17010.1 (1), 32720.1 (1), 34620.1 (1).
Glyma08g	05580.3 (1), 06000.1 (1), 09210.1 (1), 19070.1 (1), 19500.1 (1), 33510.1 (1), 40690.1 (1), 42540.1 (1), 42910.2 (1).
Glyma09g	03000.1 (1), 04340.1 (1), 23000.1 (1), 28920.1(1), 36480.1 (1), 36940.1 (1), 38130.1 (1).
Glyma10g	03470.1 (1), 04290.1 (1), 07310.1 (1), 12660.1 (1), 31230.1 (1), 34530.2 (1), 35640.1 (1), 36790.1 (1), 38330.1 (1), 41000.1 (1), 42830.1 (1).
Glyma11g	02920.1 (1), 06270.1 (1), 06900.1 (1), 11840.1 (1), 13770.1 (1), 17120.1 (1), 18300.1 (1).
Glyma12g	00770.1 (1), 05790.1 (1), 05820.1 (1), 06130.1 (1), 09980.1 (1), 11070.1 (1), 13560.1 (1), 28950.1 (1), 29960.1 (2), 32450.1 (1), 35120.1 (1), 36100.1 (1).
Glyma13g	09370.1 (1), 10410.1 (1), 10560.1 (1), 17710.1 (1), 20850.1 (1), 23350.1 (1), 27300.1 (1), 38750.1 (1), 39600.1 (1), 39930.1 (1).
Glyma14g	00940.1 (1), 22180.1 (1), 24590.1 (1), 28740.1 (1), 36580.1 (1), 39420.1 (1), 39450.1 (1), 39790.1 (1).
Glyma15g	01550.1 (1), 01550.2 (1), 03620.1 (1), 04200.1 (1), 05090.1 (1), 05470.1 (1), 18580.1 (1), 33290.1 (1), 35960.1 (1), 37830.1 (1), 40430.1 (1).
Glyma16g	02290.1 (1), 17450.1 (1), 18390.1 (1), 25110.1 (1), 29760.1 (1), 32730.1 (1).
Glyma17g	01940.1 (1), 07250.1 (1), 18890.1 (1), 22650.1 (1), 31360.1 (1), 37910.1 (1), 38040.1 (1).
Glyma18g	01950.1 (2), 05800.1 (1), 07700.1 (1), 19720.1 (1), 33480.1 (1), 33910.1 (1), 48520.1 (1), 53420.1 (1), 54010.1 (1).
Glyma19g	02330.1 (1), 02340.1 (1), 02380.1 (1), 02890.1 (1), 24870.1 (1), 27360.1 (1), 29750.1 (1).
Glyma20g	02370.1 (1), 14280.1 (1), 25140.1 (1), 26980.1 (1).

VI. CONCLUSION AND FUTURE WORK

Despite the *in silico* methodology employed in this work, our results suggest that there are miRNAs in plant transcript introns (as is also shown in [30]).

As future work, biological experiments with sequences obtained as pre-miRNA will be conducted to discover the mature miRNA and to identify the stage of plant development in which it is expressed.

Also it is intended to identify miRNAs targets in order to establish the role of miRNA in gene expression organism and to apply the gained knowledge to make plants more resistant to biotic and abiotic stresses like drought, rust, pathogens viruses and flooding.

ACKNOWLEDGMENT

The authors would like to thank Evandro Bacarin and Lucas Sampaio for their help with the English revision of this paper. This work was supported by Brazilian Soybean Genome Consortium (CNPq/GENOSOJA, grant #552735/2007-8), Accelerated Growth Programme of Embrapa (PAC Embrapa), Araucária Foundation and Coordination for the Development of Graduate and Academic Research (CAPES).

REFERENCES

[1] Conab, "Acompanhamento da safra brasileira: grãos, oitavo levantamento," Companhia Nacional de Abastecimento, Brasília, Tech. Rep., May 2010. [Online]. Available: http://www.conab.gov.br/conabweb/download/safra/8graos_6.5.10.pdf

[2] Z. Ghosh, J. Chakrabarti, and B. Mallick, "mirnomics - the bioinformatics of microrna genes," *Biochemical and Biophysical Research Communications*, vol. 363, pp. 6–11, November 2007.

[3] B. Zhang, X. Pan, and E. J. Stellwag, "Identification of soybean microrna and their targets," *Planta*, vol. 229, pp. 161–182, September 2008.

[4] N. D. Mendes, A. T. Freitas, and M. F. Sagot, "Current tools for the identification of mirna genes and their targets," *Nucleic Acids Research*, vol. 37, pp. 2419–2433, 2009.

[5] M. Yousef, L. Showe, and M. Showe, "A study of microrna in silico and in vivo: bioinformatics approaches to microrna discovery and target identification," *The FEBS Journal*, vol. 276, pp. 2150–2156, 2009.

[6] A. Sewer, N. Paul, P. Landgraf, A. Aravin, S. Pfeffer, M. Brownstein, T. Tuschl, E. van Nimwegen, and M. Zavolan, "Identification of clustered microrna using an ab initio prediction method," *BMC Bioinformatics*, vol. 6, pp. 267+, 2005.

[7] S. N. K. Loong and S. K. Mishra, "De novo svm classification of precursor microrna from genomic pseudo hairpins using global and intrinsic folding measures," *Structural Bioinformatics*, vol. 23, no. 11, pp. 1321–1330, January 2007.

[8] B. H. Zhang, X. P. Pan, S. B. Cox, G. P. Cobb, and T. A. Anderson, "Evidence that mirnas are different from others rnas," *Cell Mol Life Sciences*, pp. 246–254, January 2006.

[9] J.-W. Nam, K.-R. Shin, J. Han, Y. Lee, V. N. Kim, and B.-T. Zhang, "Human microrna prediction through a probabilistic co-learning model of sequence and structure," *Nucl. Acids Res.*, vol. 33, no. 11, pp. 3570–3581, June 2005.

[10] C. Xue, F. Li, T. He, G.-P. Liu, Y. Li, and X. Zhang, "Classification of real and pseudo microrna precursors using local structure-sequence features and support vector machine," *Bioinformatics*, vol. 6, no. 310, December 2005.

[11] M. Yousef, M. Nebozhyn, H. Shatkay, S. Kanterakis, L. C. Showe, and M. K. Showe, "Combining multi-species genomic data for microrna identification using naive bayes classifier," *Bioinformatics*, vol. 22, no. 11, pp. 1325–1334, 2006.

[12] J. Hertel and P. F. Stadler, "Hairpins in a haystack: recognizing microrna precursors in comparative genomics data," *Bioinformatics*, vol. 22, no. 14, pp. e197–e202, 2006.

[13] Z. Zhang, J. Yu, D. Li, Z. Zhang, F. Liu, X. Zhou, T. Wang, Y. Ling, and Z. Su, "Pmrd: plant microrna database," *Nucl. Acids Res.*, vol. 38, no. suppl_1, pp. D806–813, January 2010.

[14] J.-H. Yang, P. Shao, H. Zhou, Y.-Q. Chen, and L.-H. Qu, "deepbase: a database for deeply annotating and mining deep sequencing data," *Nucl. Acids Res.*, vol. 38, no. suppl_1, pp. D123–130, January 2010.

[15] J. Schmutz, S. B. Cannon, J. Schlueter, J. Ma, T. Mitros, W. Nelson, D. L. Hyten, Q. Song, J. J. Thelen, J. Cheng, D. Xu, U. Hellsten, G. D. May, Y. Yu, T. Sakurai, T. Umezawa, M. K. Bhattacharyya, D. Sandhu, B. Valliyodan, E. Lindquist, M. Peto, D. Grant, S. Shu, D. Goodstein, K. Barry, M. Futrell-Griggs, B. Abernathy, J. Du, Z. Tian, L. Zhu, N. Gill, T. Joshi, M. Libault, A. Sethuraman, X.-C. Zhang, K. Shinozaki, H. T. Nguyen, R. A. Wing, P. Cregan, J. Specht, J. Grimwood, D. Rokhsar, G. Stacey, R. C. Shoemaker, and S. A. Jackson, "Genome sequence of the palaeopolyploid soybean," *Nature*, vol. 463, pp. 178–183, January 2010.

[16] S. N. K. Loong and S. K. Mishra, "Unique folding of precursor micrornas: Quantitative evidence and implications for de novo identification," *RNA*, vol. 13, pp. 170–187, 2006.

[17] I. L. Hofacker, W. Fontana, P. F. Stadler, L. S. Bonhoeffer, M. Tacker, and P. Schuster, "Fast folding and comparison of rna secondary structures," *Monatshefte fúr Chemie*, vol. 125, pp. 167–188, 1994.

[18] J. Reedera, M. Hochsmanna, M. Rehmsmeierb, B. Vossc, and R. Giegerich, "Beyond mfold: Recent advances in rna bioinformatics," *Journal of Biotechnology*, vol. 124, no. 1, pp. 41–55, June 2006.

[19] E. A. Schultes, P. T. Hraber, and T. H. LaBean, "Estimating the contributions of selection and self-organization in rna secondary structure," *Journal of Molecular Evolution*, vol. 49, no. 1, pp. 76–83, July 1999.

[20] E. Freyhult, P. Gardner, and V. Moulton, "A comparison of rna folding measures," *BMC Bioinformatics*, vol. 6, no. 1, pp. 241+, October 2005.

[21] A. Gruber, S. Bernhart, I. Hofacker, and S. Washietl, "Strategies for measuring evolutionary conservation of rna secondary structures," *BMC Bioinformatics*, vol. 9, no. 1, pp. 122+, February 2008.

[22] H. H. Gan, D. Fera, J. Zorn, N. Shiffeldrim1, M. Tang, U. Laserson, N. Kim, and T. Schlick, "Rag: Rna-as-graphs databaseconcepts, analysis, and features," *Bioinformatics*, vol. 20, no. 8, p. 12851291, February 2004.

[23] S. N. K. Loong and S. K. Mishra, "Unique folding of precursor micrornas: Quantitative evidence and implications for de novo identification," *RNA Supplementary materials*, vol. 13, 2006.

[24] W. S. Noble, "Support vector machine applications in computational biology," in *Kernel methods in computational biology*, B. Scholkopf, K. Tsuda, and J.-P. Vert, Eds. Cambridge, MA: MIT Press, 2004, ch. 3.

[25] S. Haykin, *Redes Neurais: princípios e práticas*. Porto Alegre: Bookman, 1999.

[26] J. C. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods - Support Vector Learning*, B. Scholkopf, C. J. C. Burges, and A. J. Smola, Eds. Cambridge, MA: MIT Press, 1999, pp. 41–65.

[27] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, "A practical guide to support vector classication," April 2010. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf

[28] I. Bentwich, "Prediction and validation of microrna and their targets," *FEBS Letters*, vol. 579, pp. 5904–5910, 2005.

[29] S. Altschul, W. Gish, W. Miller, E. Myers, and D. Lipman, "Basic local alignment search tool," *J. Mol. Biol.*, vol. 215, pp. 403–410, 1990.

[30] J. W. S. Brown, D. F. Marshall, and M. Echeverria, "Intronic noncoding rnas and splicing," *Trends in Plant Science*, vol. 13, no. 7, pp. 335–342, June 2008.