YURI BENTO MARQUES

MIRNACLE: APRENDIZAGEM DE MÁQUINA UTILIZANDO SMOTE E RANDOM FOREST PARA PROVER AUMENTO DA SELETIVIDADE NA PREDIÇÃO AB INITIO DE PRE-MIRNAS

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae*.

VIÇOSA
MINAS GERAIS - BRASIL
2015

Ficha catalográfica preparada pela Biblioteca Central da Universidade Federal de Viçosa - Câmpus Viçosa

T
M357m
2015

Marques, Yuri Bento, 1984-
Mirnacle : aprendizagem de máquina utilizando SMOTE e Random Forest para prover aumento da seletividade na predição ab initio de pre-miRNAs / Yuri Bento Marques. – Viçosa, MG, 2015.
xiii, 87f. : il. (algumas color.) ; 29 cm.

Inclui apêndices.
Orientador: Fábio Ribeiro Cerqueira.
Dissertação (mestrado) - Universidade Federal de Viçosa.
Referências bibliográficas: f.49-54.

1. Aprendizado do computador. 2. Bioinformática. 3. Biologia molecular. 4. Ácido ribonucleico. I. Universidade Federal de Viçosa. Departamento de Informática. Programa de Pós-graduação em Ciência da Computação. II. Título.

CDD 22. ed. 006.31

YURI BENTO MARQUES

MIRNACLE: APRENDIZAGEM DE MÁQUINA UTILIZANDO SMOTE E RANDOM FOREST PARA PROVER AUMENTO DA SELETIVIDADE NA PREDIÇÃO AB INITIO DE PRE-MIRNAS

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae.*

APROVADA: 08 de dezembro de 2015.

Ricardo dos Santos Ferreira

Sabrina de Azevedo Silveira

Elizabeth Pacheco Batista Fontes

Fábio Ribeiro Cerqueira
Orientador

Aos meus pais, Francisco de Assis Marques e Maria Cristina Bento Marques
Ao meu irmão Francis,
À minha esposa Patrícia,
À minha filha Luna

"Qualquer tecnologia suficientemente avançada não se distingue de mágica"
(Arthur Charles Clarke)

AGRADECIMENTOS

A Deus, em primeiro lugar, por tudo.

Aos meus pais, meu irmão, minha esposa e minha filha pelo apoio incondicional.

Aos meus amigos, em especial meu Master Felipe, sem você não teria começado e tão pouco, terminado :)

Aos meus colegas do DPI, em especial Adílson, André, Cleydson, Cristiane, Dênis, Gerardo, Gilberto, Ítalo, Marcos, Marques, Néliton, Nilson, Robert, Thales e Willian, muito obrigado pela parceria!

Ao meu orientador, Prof. Fábio Cerqueira, pela confiança, paciência e tantos ensinamentos, MUITO obrigado!

Aos servidores da UFV, em especial Altino, Prof. André, Prof. Alcione, Prof. Lucas, Prof. Mauro, Prof. José Luís, Prof. Jugurta e Prof. Mauro, muito obrigado por nos proporcionar um ambiente tão fantástico que é a UFV!

Aos meus professores na graduação, em especial Profs. Alcilene, Ciro, Daniel, Edimar, Henrique, Kennedy, Mardem e Renato, sem o ensinamento de vocês, não teria chegado até aqui, obrigado!

Aos meus colegas do IFNMG, em especial Adriana, André, Ana Flávia, Alyson, Heleno, Henrique, Gama, Joan, José Rui (E sua família sensacional :D), Paulo, Pedro, Jeferson, Renildo, Roberta, Rodrigo e Thiago, agradeço imensamente o suporte para este trabalho, obrigado!

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Siglas

ANN - *artifical neural network* - rede neural artificial

DNA - ácido desoxirribonucleico

FN - falso negativo

FP - falso positivo

KKT - *karush-kuhn-tucker*

miRNAs - microRNAs

miscRNAs - *miscellaneous RNA*

MLP - *multilayer perceptron*

mRNA - RNA mensageiro

nt - nucleotídeos

pre-miRNA - microRNA precursor

RF - *Random Forests*

RISC - Complexo de Indução do Silenciamento do RNA

RNA - ácido ribonucleico

rRNA - RNA ribossômico

SCFG - *Stochastic Context Free Grammar*

siRNA - *short interfering RNA*

SMO - *Sequential Minimal Optimization*

SMOTE - *Synthetic Minority Over-sampling Technique*

snoRNA - *small nucleolar RNA*

SNP - Polimorfismo de nucleotídeo único

snRNA - *small nuclear RNA*

SVM - máquina de vetores de suporte

TP - verdadeiro positivo

tRNA - RNA Transportador

Resumo

MARQUES, Yuri Bento, M.Sc., Universidade Federal de Viçosa, dezembro de 2015. **Mirnacle: Aprendizagem de máquina utilizando SMOTE e Random Forest para prover aumento da seletividade na Predição ab initio de pre-miRNAs** Orientador: Fábio Ribeiro Cerqueira. Coorientador: Alcione de Paiva Oliveira.

Os microRNAs (miRNAs) são importantes reguladores da expressão gênica em plantas e animais. Assim, miRNAs estão envolvidos na maioria dos processos biológicos, tornando o estudo dessas moléculas um dos temas mais relevantes da biologia molecular atualmente. Uma estratégia para encontrar novos miRNAs é procurar seus precursores (pre-miRNAs), que são estruturas ligeiramente maiores (70-120 nt) e têm uma estrutura secundária na forma de *hairpin* (grampo de cabelo). No entanto, caracterizar pre-miRNAs *in vivo* ainda é uma tarefa complexa. Como consequência disto, métodos *in silico* foram desenvolvidos para prever a localização genômica de pre-miRNAs. No entanto, as ferramentas computacionais atuais têm problemas de seletividade, isto é, uma grande quantidade de falsos positivos é reportada. Este trabalho apresenta uma extensão do método desenvolvido por Tempel e Tahi, 2012, com o objetivo de melhorar a seletividade através da técnica de aprendizagem de máquina denominada Random Forest, combinada com o método SMOTE, que lida com conjuntos de dados desbalanceados. Comparando o método proposto com outras importantes abordagens na literatura, mostramos que os procedimentos descritos neste trabalho puderam melhorar substancialmente a seletividade, sem comprometer a sensibilidade. Para três conjuntos de dados utilizados nos experimentos realizados, a abordagem proposta alcançou pelo menos 97 % de sensibilidade e proporcionou um aumento de duas, vinte e seis vezes na seletividade, respectivamente, em comparação com os resultados de ferramentas computacionais atuais.

Abstract

MARQUES, Yuri Bento, M.Sc., Universidade Federal de Viçosa, December, 2015. **Mirnacle: Machine learning with SMOTE and random forest for improving selectivity in pre-miRNA ab initio prediction** Adviser: Fábio Ribeiro Cerqueira. Co-adviser: Alcione de Paiva Oliveira

MicroRNAs (miRNAs) are key gene expression regulators in plants and animals. Thus, miRNAs are involved in the majority of biological process, making the study of these molecules one of the most relevant topics of molecular biology nowadays. A strategy to find new miRNAs is to search for its precursors (pre-miRNAs), which are slightly larger structures (70-120 nt) and have a hairpin structural form. However, characterizing pre-miRNAs in vivo is still a complex task. As a consequence, in silico methods were developed to predict the genomic location of pre-miRNAs. Nevertheless, the current computational tools have problems of selectivity, i.e., a higher number of false positives is reported. This work presents an extension of the method developed by Tempel and Tahi, 2012, with the aim of improving selectivity through machine learning techniques, namely, random forests combined with the SMOTE method that copes with imbalance datasets. Comparing our method with other important approaches in the literature, we have shown that our procedures could substantially improve selectivity without compromising sensibility. For three datasets used in our experiments, our method achieved at least 97% of sensitivity and could deliver a two-fold, 20-fold, and 6-fold increase in selectivity, respectively, compared with the best results of current computational tools.

Capítulo 1

Introdução

Estudar biologia significa conhecer os seres vivos, ou seja, entender sobre nós mesmos e nossa subsistência. Por consequência disto, é importante a utilização de tecnologias para simplificar e auxiliar no entendimento dos complexos sistemas biológicos [Thangadurai & Sangeetha, 2014].

Os marcos mais importantes na genética e genômica começaram com a descoberta das leis de hereditariedade nos primórdios do século XX. O reconhecimento do DNA como o material hereditário, a determinação de sua estrutura, elucidação do código genético, desenvolvimento de tecnologias de DNA recombinante e o estabelecimento de métodos cada vez mais automatizados de sequenciamento de DNA serviram como suporte para a criação do Projeto Genoma Humano [Collins et al., 2003].

As ciências biomédicas e biológicas estão ficando cada vez mais interdisciplinares, o que demanda dos futuros cientistas, treinamentos interdisciplinares ao invés da formação disciplinar convencional. Um dos fundadores da biologia molecular, Francis Crick (1965) se considerava uma mistura de biofísico, bioquímico, cristalografista e geneticista. Em resposta às demandas interdisciplinares das ciências biomédicas e biológicas, foi criada a Bioinformática. Ela nasceu dos esforços conjuntos de matemáticos, cientistas da computação e biólogos [Xia, 2007].

A combinação de ferramentas computacionais, bases de dados de conhecimento integrado e as abordagens tradicionais de biologia, bioquímica, química, física, matemática e genética, aumenta a esperança de compreender a função e regulação de todos os genes e proteínas, decifrar as atividades fundamentais das células, determinar os mecanismos de funcionamento das doenças e descobrir maneiras de intervirem com ou impedir processos celulares anormais, a fim de melhorar a saúde humana e o bem-estar [Lockhart & Winzeler, 2000].

1.1 O problema e sua importância

MicroRNAs (miRNAs) são reconhecidos como reguladores chave da expressão gênica em plantas e animais. MiRNAs são pequenos RNAs não codificantes (aproximadamente 22 nt) que regulam a expressão gênica se ligando a um RNA mensageiro (mRNA) alvo e promovendo sua degradação ou inibição da tradução [Bartel, 2004].

Os MicroRNAs estão envolvidos em diversos processos biológicos e fisiológicos, tais como: diferenciação embrionária [Suh et al., 2004], desenvolvimento de músculos [Williams et al., 2009] e esqueleto [Chen et al., 2005], hematopoese [Shivdasani, 2006], controle da morte e proliferação das células [Ambros, 2004] [Brennecke et al., 2003], secreção de insulina [Poy et al., 2004], metabolismo dos lipídios [Wilfred et al., 2007] e doenças, como o câncer [Murchison et al., 2007] [Osada & Takahashi, 2007], o que ressalta sua importância na biologia molecular atualmente [Zare et al., 2014].

Em sua biogênese, os miRNAs são concebidos através da transcrição de seu gene em um miRNA primário (pri-miRNA), que após clivagem pela enzima Drosha resulta em um precursor de miRNA (pre-miRNA) de aproximadamente 70 nucleotídeos (nt), o qual possui formato de *hairpin* (grampo de cabelo). Por fim, o pre-miRNA é exportado para o citoplasma pela proteína Exportin5 e clivado pela enzima Dicer para produzir um miRNA maduro [Krol et al., 2010] [Ha & Kim, 2014].

Identificar a localização de um novo gene de microRNA *in vivo* é uma tarefa cara e complexa. Assim, ferramentas computacionais têm sido desenvolvidas para predizer a localização dos mesmos *in silico*. Diferentemente das abordagens comparativas, os métodos ab initio são capazes de trabalhar com genomas que não possuem espécies relacionadas [Tempel & Tahi, 2012]. Os chamados métodos completamente ab initio recebem um genoma e não utilizam outras informações biológicas para localizar novos pre-miRNAs. Porém, as ferramentas computacionais disponíveis no presente momento têm problemas de seletividade, o que compromete a validação *in vivo* dos resultados gerados. Portanto, é necessário o desenvolvimento de novas ferramentas computacionais para predição de pre-miRNAs com melhor acurácia [Tempel & Tahi, 2012].

1.2 Hipótese

Com o uso de técnicas de aprendizagem de máquina é possível construir um método computacional que melhore a seletividade na predição ab initio de pre-miRNAS, sem comprometer a sensibilidade.

1.3 Objetivo

O objetivo deste trabalho é desenvolver um método computacional completamente ab initio de predição de novos pre-miRNAs em *Homo sapiens* e *Mus musculus*, que mantenha a sensibilidade das ferramentas atuais, porém, com uma melhor seletividade.

Para que o objetivo seja alcançado, será desenvolvida uma extensão do método proposto por Tempel & Tahi [2012] e co-autores, adicionando técnicas de aprendizagem de máquina visando ganhos de seletividade.

Por fim, pretende-se disponibilizar o método em uma ferramenta computacional de fácil utilização, desenvolvida em tecnologias livres visando prover a comunidade científica acesso aos resultados e melhoramentos do método proposto.

1.4 Organização do Trabalho

Em razão do trabalho ter como linha de pesquisa a bioinformática, no Capítulo 2 são apresentados os principais conceitos biológicos referentes a esta pesquisa, assim como os algoritmos de aprendizagem de máquina utilizados. Adicionalmente, no mesmo capítulo também são apresentados os métodos existentes na literatura que visam solucionar o problema proposto. Em seguida, no Capítulo 3, é apresentado o método proposto neste trabalho para predição ab initio de pre-miRNAs. Em consequência disto, no Capítulo 4 são apresentados os resultados do método proposto, realizadas discussões e comparações deste trabalho frente ao estado da arte. Por fim, no Capítulo 5 são mostradas as conclusões, contribuições e possíveis trabalhos futuros resultantes desta pesquisa.

Capítulo 2

Revisão Bibliográfica

2.1 Conceitos em biologia molecular

2.1.1 Genoma

De acordo com o dicionário Oxford English dictionary, o termo *genom(e)* foi cunhado por Hans Winkler em 1920 como junção de *gene* e *chromosome*. A palavra genoma pode ser utilizada para se referenciar o conjunto completo dos genes ou a quantidade total do DNA em um conjunto de cromossomas haploide [Gregory, 2011].

Todas as criaturas têm seu próprio genoma, porém a estrutura básica do material genético é a mesma. Por consequência disto, os ratos são bastante utilizados em testes nas pesquisas científicas, devido à estrutura do genoma do rato ter muita semelhança com a estrutura do genoma humano [Fridell, 2005].

Não basta apenas identificar as partes componentes do genoma e simplesmente montar um catálogo de todos os genes e suas funções. Além disto, é necessário utilizar os métodos computacionais e experimentais para obter o máximo de informação das sequências, entender como os componentes trabalham em conjunto para que seja possível compreender o funcionamento completo das células e organismos [Lockhart & Winzeler, 2000].

2.1.2 Genômica

A Genômica é o estudo de todos os genes de um organismo, as interações entre genes e as interações dos genes com o ambiente. Desta forma, é possível investigar doenças como diabetes, asma, câncer e problemas no coração, pois essas doenças podem ser causadas pela combinação de fatores genéticos e ambientais. Por fim, novas formas de

tratamento, terapia e diagnósticos podem ser desenvolvidos para aqueles que sofrem com estas patologias.

2.1.3 DNA

Por pelo menos parte da vida de um ser vivo, todas as células têm pelo menos um núcleo ou nucleoide, no qual o genoma (o conjunto completo dos genes) é armazenado e replicado. Nas bactérias e archaea, o nucleoide não é separado do citoplasma por uma membrana. Já em eucariotos, o núcleo consiste de material nuclear enclausurado dentro de uma membrana dupla, o envelope nuclear [Nelson et al., 2008].

O ácido nucleico (DNA) é construído de monômeros de nucleotídios (nt) que consistem de açúcar (desoxirribose) e um grupo fosfato compreendendo o esqueleto do biopolímero junto com uma de quatro bases nitrogenadas representadas pelas letras do alfabeto: A (Adenina), T (Timina), C (Citosina) e G (Guanina) [Gregory, 2011].

O DNA pode ser transcrito em ácido ribonucleico (RNA), o qual consiste dos nucleotídios A, G, C e U (Uracila) [Pevsner, 2009].

A Figura 2.1 (a) mostra um modelo tridimensional da molécula de DNA. Além disto, na Figura 2.1 (b) um diagrama da mesma molécula de DNA com os arranjos açúcar-fosfato e os pares de base. Por fim, a Figura 2.1 (c) mostra a estrutura química da dupla-hélice de DNA [Russell, 2010].

2.1.4 RNA

O RNA é em sua composição primária, similar ao DNA, diferindo em ter açúcar ribose em vez de desoxirribose e uracila como base de pirimidina ao invés de timina [Russell, 2010].

O processo de transcrição de DNA resulta na formação de moléculas de RNA de duas grandes classes. A primeira contém os RNAs codificantes, formados quando um DNA é transcrito em um RNA mensageiro (mRNA). Posteriormente, este mRNA é traduzido em proteína por meio de um processo mediado por RNAs transportadores (tRNA) e RNAs ribossômicos (rRNA), assim como por proteínas. A outra grande classe de RNAs são os RNAs não codificantes os quais são produtos da transcrição do DNA, porém, não são traduzidos em proteínas [Pevsner, 2009].

A Figura 2.2 mostra uma ilustração do DNA e RNA. Enquanto DNA geralmente adota uma conformação helicoidal dupla, RNA tende a ter uma fita única. As exceções são os RNAs não codificantes, que podem ter formações como *stem-loop* [Pevsner, 2009].

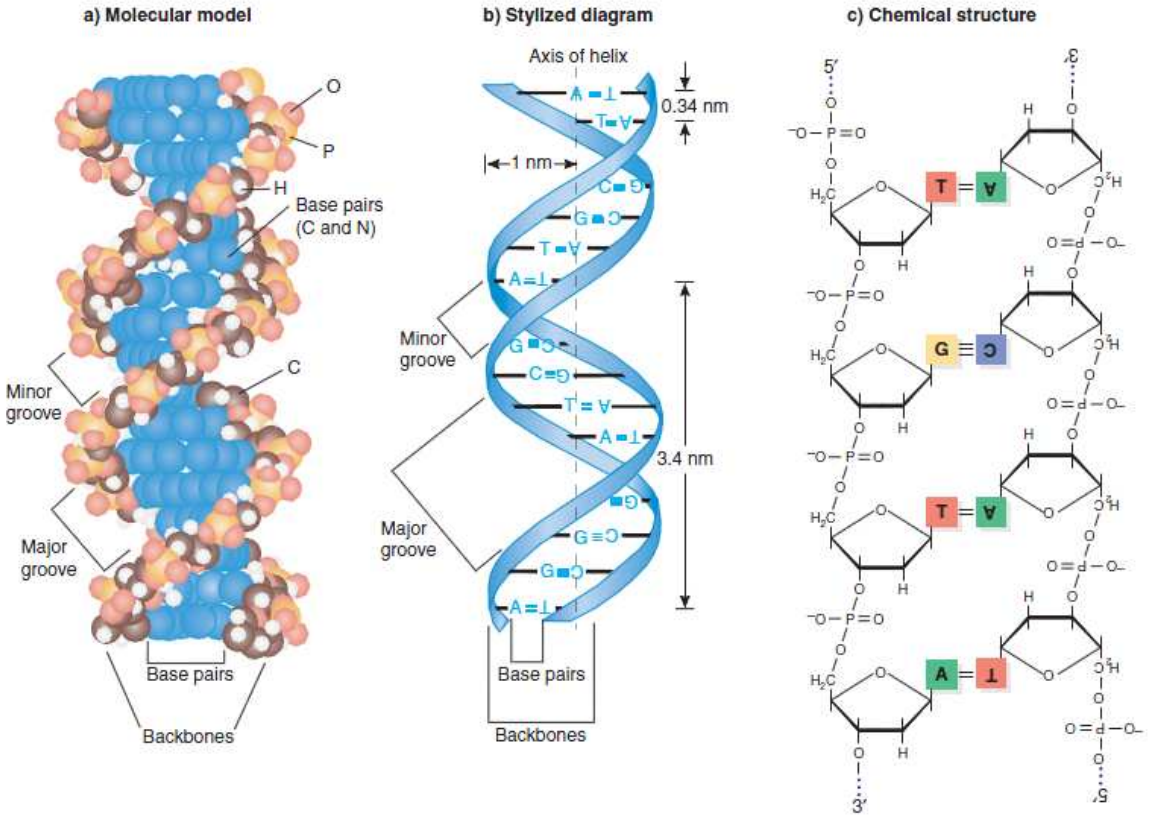

Figura 2.1. Estrutura molecular do DNA. Fonte: Russell [2010]

Para determinar a estrutura secundária de um RNA baseado em sua sequência de nucleotídeos, configurações de pares de bases são testadas e a energia livre é calculada a partir de informações como: quantidade de pares de bases, GC, AU, GU, número de pares de bases nas regiões da haste, número de bases não pareadas. A configuração que resultar em uma menor energia livre é chamada de configuração de energia livre mínima. Como resultado final, esta configuração é normalmente considerada a provável estrutura secundária da molécula de RNA [Nelson et al., 2008].

Vários esforços têm sido realizados para realizar a predição *in silico* da estrutura secundária de um dado RNA. Como exemplo, podemos citar Vienna RNAfold [Hofacker, 2003], mFold [Zuker, 2003] e GaRNA [Titov et al., 2002]. A Figura 2.3 mostra a sequência de RNA GGGCUAUUAGCUCAGUUGGUUAGAGCGCACCCCUGAUAAGGGUGAGGUCGCUGAUUCGAAUUCAGCAUAGCCCA sendo analisada por estas três ferramentas, utilizando os parâmetros padrões.

Neste exemplo, Vienna RNAfold e GaRNA apresentaram a mesma estrutura. Por outro lado, o mFold apresentou uma estrutura diferente, conforme evidenciado na notação *dot bracket* mostrada na Figura 2.4. Percebe-se, portanto, que os resultados

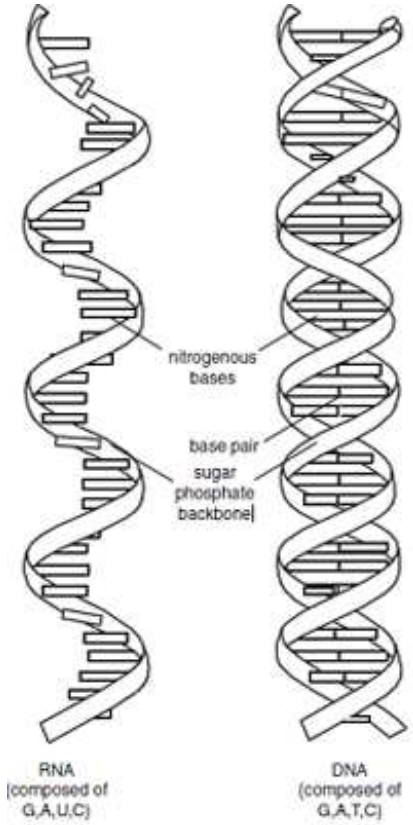

Figura 2.2. Ácido desoxirribonucleico (DNA) e ácido ribonucleico (RNA). Diferenças na conformação das moléculas Fonte: Pevsner [2009]

.

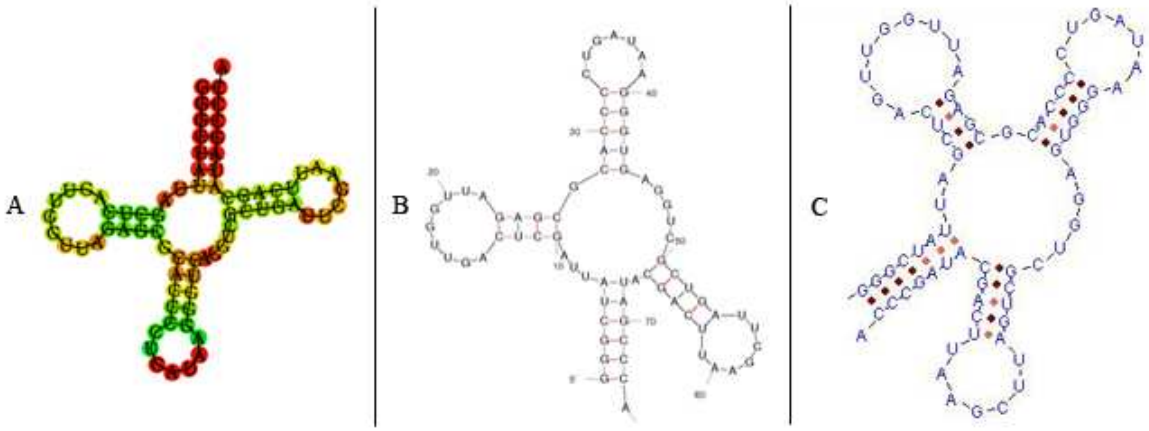

Figura 2.3. Predição da estrutura secundária de um RNA. A) Vienna RNAfold B) mFold C) GaRNA

diferentes podem comprometer a pesquisa, se forem utilizadas mais de uma ferramenta de predição no mesmo trabalho. Sendo assim, nesta pesquisa utilizamos o Vienna

RNAfold para todos os cálculos de energia livre mínima.

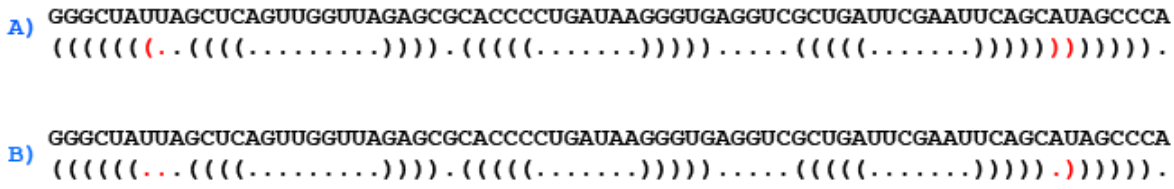

Figura 2.4. Diferença na predição da estrutura secundária de um determinado RNA. A) Vienna RNAfold e GaRNA B) mFold

.

2.1.5 Aminoácidos

No processo de tradução, os nucleotídeos do RNA mensageiro são lidos de três em três (triplets). Ex: AAA, AUC, UUU, UAC, etc. Estes três nucleotídeos são chamados de códon. Cada sequência de códons representa um aminoácido ou sinal de parada do processo de tradução Alberts et al. [2010].

A Figura 2.5 ilustra o processo de tradução de nucleotídeos em aminoácidos.

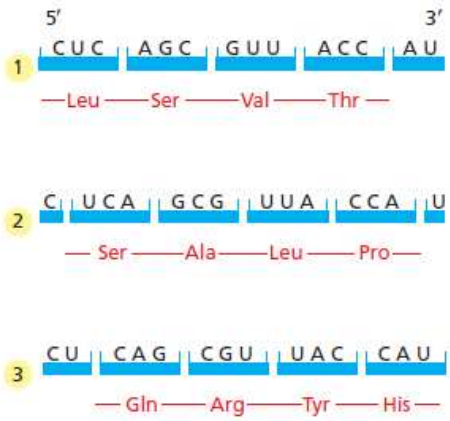

Figura 2.5. Ilustração da tradução de nucleotídeos (códons de cor azul) em aminoácidos (representados em vermelho). Fonte: Alberts et al. [2010]

.

Os aminoácidos são caracterizados pela mesma estrutura central, a qual pode ser ligada a qualquer outro aminoácido através de ligações denominadas peptídicas. Sendo assim, as proteínas são sintetizadas através da união dos aminoácidos em estruturas de três dimensões [Alberts et al., 2010].

2.1.6 Proteínas

As proteínas são longos polímeros de aminoácidos, que constituem a fracção maior (além de água) de uma célula. Elas podem ter atividade catalítica e funcionar como enzimas. Também podem servir como elementos estruturais, sinalizar os receptores, ou transportadores que carregam substâncias específicas dentro ou fora das células. Devido as suas várias funções, as proteínas podem ser consideradas a mais versátil de todas as biomoléculas. Além disto, o conjunto de proteínas funcionando em uma célula é chamado de Proteoma [Nelson et al., 2008]. A Figura 2.6 ilustra a tradução do RNA em proteína.

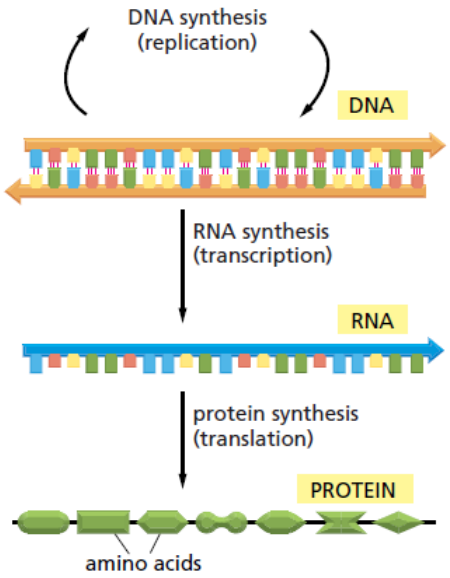

Figura 2.6. Ilustração dos processos de replicação, transcrição e tradução. Fonte: Alberts et al. [2010]

.

complemento

2.1.7 MicroRNAs

As principais classes de RNAs não codificantes são tRNA e rRNA, os quais juntos respondem por aproximadamente 95% de todos os RNAs. Outros importantes RNAs não codificantes são small nuclear RNA (snRNA), small nucleolar RNA (snoRNA), short interfering RNA (siRNA) e microRNA [Pevsner, 2009].

Os genes de microRNAs (miRNAs) são reconhecidos como reguladores cruciais da expressão gênica em plantas e animais. Além disto, miRNAs são pequenos (aproximadamente 22 nt) RNAs não codificantes que regulam a expressão gênica ligando-se

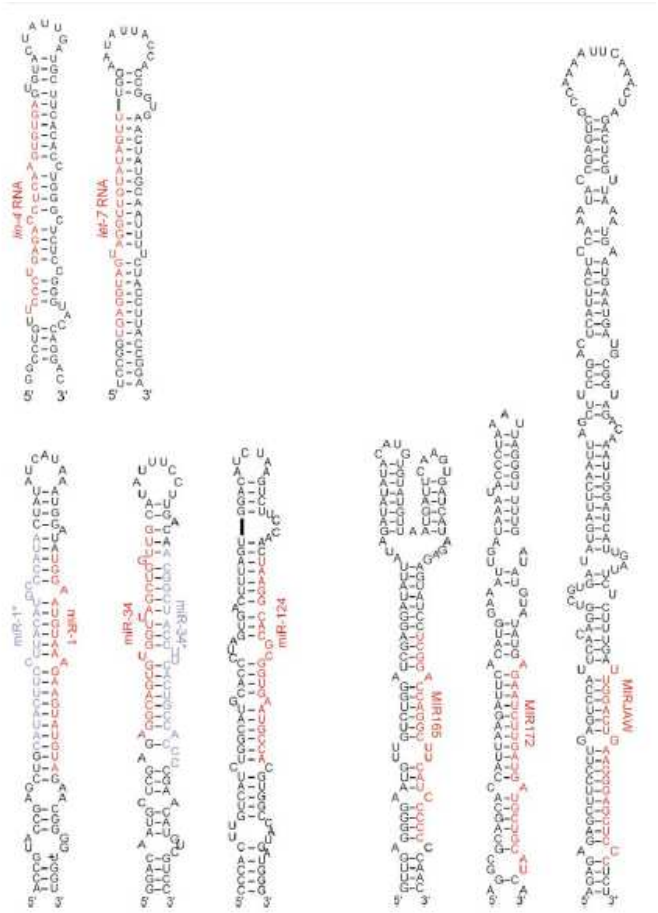

Figura 2.7. Exemplo de *Hairpins* e miRNAs (em vermelho). A estrutura conhecida como bojo é formada quando não existe ligação de complementaridade de bases em um lado da fita e estrutura de *loop* quando não existem ligações dos dois lados. Fonte: Bartel [2004]

a um específico mRNA alvo e promovendo sua degradação e ou inibição da tradução [Bartel, 2004].

Os miRNAs são transcritos primeiramente como longos miRNAs primários (pri-miRNAs) e, então, quebrados em miRNAs precursores (pre-miRNAs) pelo complexo Drosha/Pasha. O pre-miRNA, estruturado como um *hairpin* ou *stem-loop* (ilustrado na Figura 2.7), é exportado para o citoplasma pela proteína Exportin5 e, então, é quebrado pelo Dicer em miRNA maduro. Por fim, incorporado ao complexo RISC, o miRNA se vincula a um mRNA específico e acarreta em uma degradação ou inibição da tradução do mRNA [Bartel, 2004]. Este processo pode ser visualizado na Figura 2.8.

A Figura 2.9 demonstra as funções do miRNA.

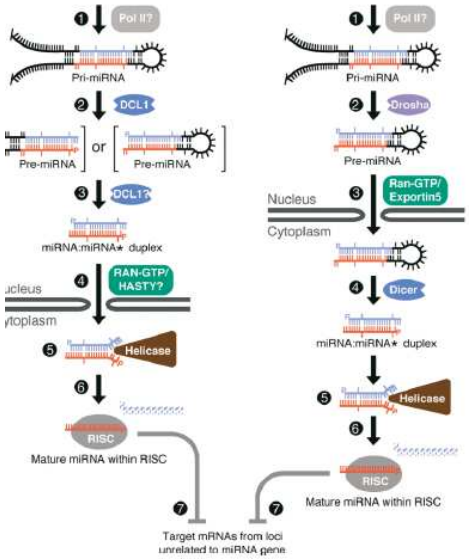

Figura 2.8. Ilustração da biogênese de miRNAs. Estruturas secundárias de pre-miRNAs (com o formato de *Hairpin*) e seus respectivos miRNAs maduros. Fonte: Bartel [2004]

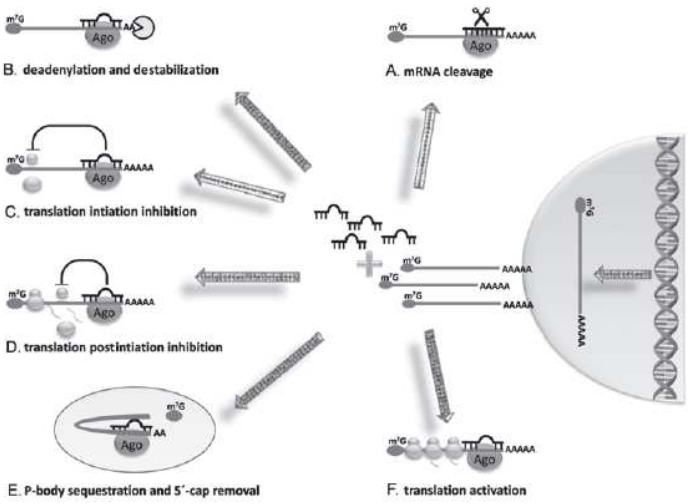

Figura 2.9. Diagrama esquemático dos mecanismos propostos para a função dos miRNAs. (A) degradação do mRNA pode ocorrer quando a sequência de miRNA é complementar ao alvo. (B) Remoção da cauda poli por deadenilase causa destabilização e degradação do mRNA. (C) Início da tradução é inibida pelo miRISC. (D) Inibição da tradução pós-iniciação. (E) Sequestro de mRNA. (F) miRNA mediando a ativação translacional. Fonte: Lawrie [2013]

2.1.8 mirBase

A biblioteca mirBase (www.mirbase.org) desenvolvida por Griffiths-Jones et al. [2006] é uma base de dados online que visa armazenar informações sobre miRNAs, como suas

sequências, anotações e predição de seus alvos. É possível buscar os microRNAs e pre-miRNAs por espécies e, então, visualizar todas suas características, assim como as suas evidências e referências.

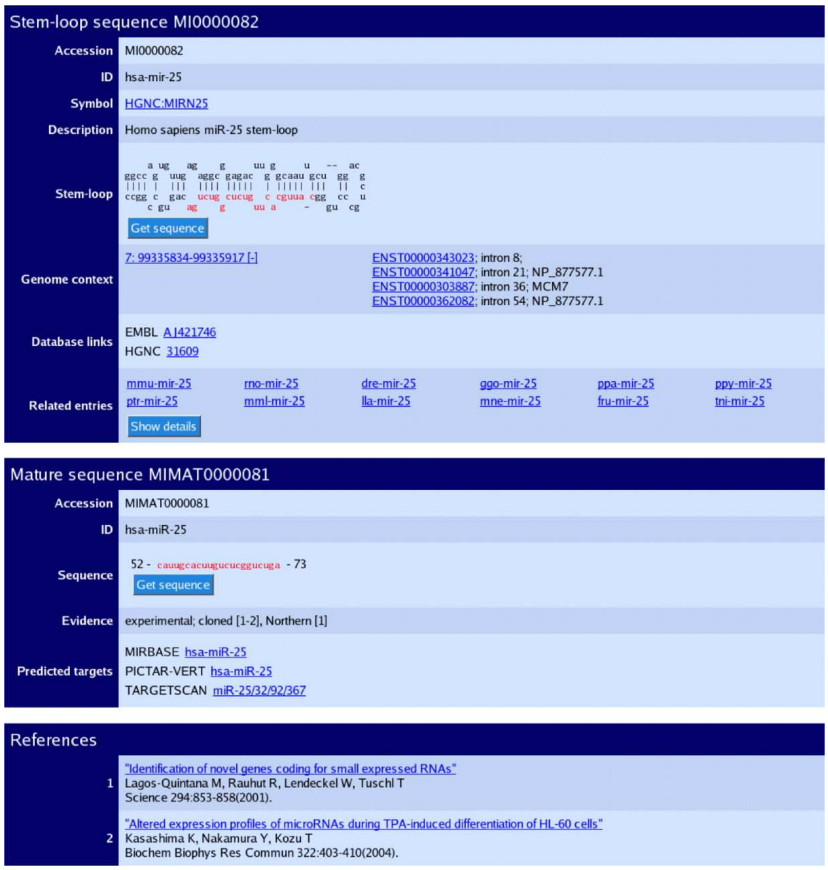

Figura 2.10. Tela do mirBase com as características do miRNA hsa-mir-25. Fonte: Griffiths-Jones et al. [2006]

2.1.9 Expressão Gênica

Expressão Gênica é o processo pelo qual a informação codificada em um gene é utilizada para controlar a produção de uma molécula de proteína ou RNA. No caso de proteínas, a célula lê a sequência do gene em grupos de três bases. Cada um dos grupos de três bases (*codon*) corresponde a um dos 20 tipos diferentes de aminoácidos utilizados para produzir proteínas [Genetics, 2013].

Na Figura 2.11 ambas regiões codificantes e não codificantes são transcritas no chamado mRNA primário. Então, as regiões não codificantes (chamadas de *introns*) são removidas durante o início do processamento de mRNA. Além disto, as partes codificantes (*exons*) restantes são unidas em conjunto e a molécula de mRNA maduro

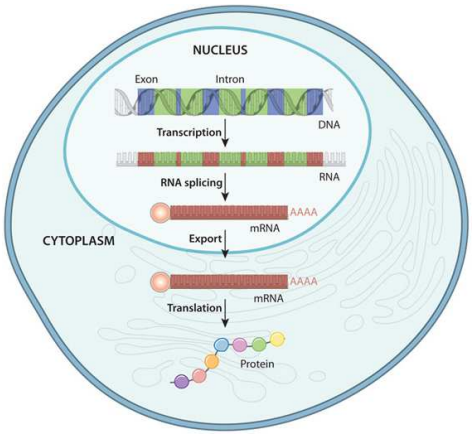

Figura 2.11. Uma visão geral sobre o fluxo de informação a partir do DNA para a proteína numa célula eucariótica. Fonte: Nature [2010]

(em vermelho) é preparada para a exportação para fora do núcleo. Por fim, quando o mRNA chegar ao citoplasma, poderá ser utilizado para produção de proteínas [Russell, 2010].

O Transcriptoma são todos os RNAs expressos em determinadas condições e isso possibilita mensurar quais genes são transcritos e quais proteínas podem ser produzidas [Russell, 2010].

2.2 Aprendizagem de Máquina

Subárea da inteligência artificial, a aprendizagem de máquina (AM) tem como objetivo principal desenvolver técnicas e algoritmos que possibilitem os computadores aprender. Este conhecimento pode ser utilizado em diversas finalidades, como: motores de busca, processamento de linguagem natural, diagnóstico médico, análise do mercado de ações, visão computacional, jogos eletrônicos, bioinformática e quimioinformática [Zhang & Rajapakse, 2009].

Na área de bioinformática, as técnicas de aprendizagem de máquina são aplicadas em problemas como: seleção de atributos, reconhecimento de promotor, predição da estrutura de proteínas, seleção de genes e de polimorfismo de nucleotídeo único (SNP), classificação e *data mining* [Zhang & Rajapakse, 2009].

A mineração de dados é um processo automático ou semiautomático de descoberta de padrões em quantidades substanciais de dados. Os padrões encontrados devem ter significado no contexto em que estão inseridos, para que seja possível a obtenção de alguma informação valiosa para o estudo em questão. Além disto, a mineração de dados é realizada através da utilização de técnicas de aprendizagem de máquina para

extrair informações em dados brutos de bases de dados [Witten & Frank, 2005].

2.2.1 Técnicas Utilizadas

De acordo com o problema a ser estudado e suas respectivas bases de dados, diferentes algoritmos de aprendizagem de máquinas podem ser utilizados. As abordagens mais utilizadas na bioinformática podem ser definidas como [Inza et al., 2010]:

- Aprendizagem Supervisionada: Dado um conjunto de treinamento com exemplos de entrada e saída conhecidos, é construído um modelo para prever a saída de futuros alvos desconhecidos. Caso os rótulos sejam um conjunto definido de valores discretos, essa ação é considerada uma classificação. Caso contrário, ou seja, um número contínuo de valores, é considerada regressão. A Figura 2.12 ilustra abordagem geral de aprendizagem supervisionada.
- Aprendizagem não Supervisionada(*clustering*): Diferentemente da aprendizagem supervisionada, os dados não contém rótulos na aprendizagem não supervisionada. O objetivo dessa técnica é dividir os dados em subconjuntos (*clusters*) de forma que os objetos de mesmo grupo tenham a maior similaridade possível e os objetos de grupos distintos sejam o mais dissimilar possível.
- Aprendizagem Semi-Supervisionada: Utiliza a mesma estratégia da aprendizagem supervisionada, porém alguns exemplos da base de treinamento tem os rótulos desconhecidos. Prediz o rótulo de novos exemplos que não se sabe a classe.

2.2.1.1 Random Forests

As árvores de classificação são muito utilizadas em problemas que necessitem classificar instâncias desconhecidas. Como vantagens desse algoritmo estão sua velocidade em relação a outros métodos e sua compreensível representação gráfica. Além disto, seu procedimento recursivo *top-down* permite que o modelo gerado seja facilmente verificado [Inza et al., 2010].

Para construir uma árvore de decisão, todo conjunto de treinamento é considerado na raiz, onde o teste de decisão que particionará os dados (normalmente em dois subconjuntos, em árvores binárias) será definido selecionando-se de forma gulosa o melhor atributo para realizar tal partição. O atributo escolhido será aquele que, de acordo com algum critério (frequentemente utiliza-se entropia), organiza melhor os dados, ou seja, um atributo que tem grande relação com a classe em questão. Este processo segue de forma recursiva para os outros nodos da árvore, utilizando a partição dos dados

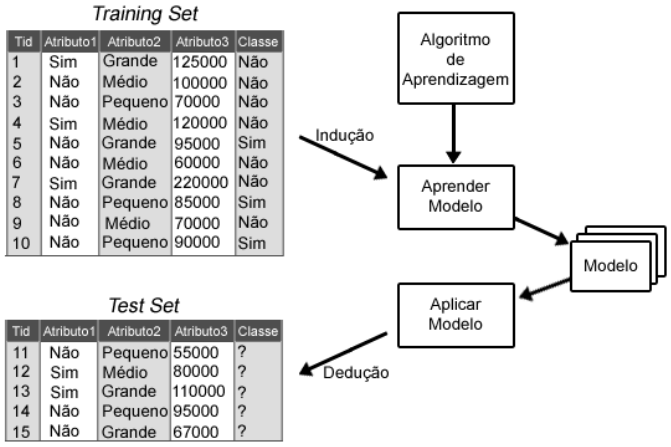

Figura 2.12. Aprendizado Supervisionado. A fim de criar o modelo, o algoritmo de aprendizagem de máquina é aplicado no conjunto de dados com rótulos conhecidos. Desta forma, é possível deduzir novas informações em conjuntos de dados com rótulos desconhecidos. Fonte: Adaptado de Pang-Ning et al. [2005]

vinda dos nodos pai, até as folhas da árvore que indicarão os rótulos da classificação. O critério de parada que permite a geração de uma folha pode ser uma partição com nenhum elemento ou com um número muito baixo de elementos. Após este processo, para classificar uma nova instância, basta seguir da raiz até uma folha de acordo com os valores dos atributos. O rótulo da folha alcançada representará a classe da instância. A Figura 2.13 ilustra o processo de classificação através de uma árvore de decisão [Inza et al., 2010].

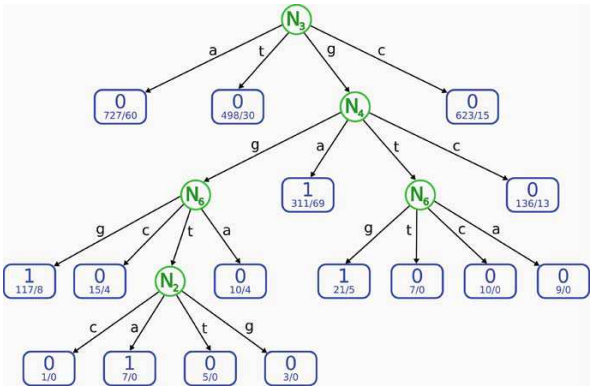

Figura 2.13. Exemplo de árvore de classificação. Fonte: Inza et al. [2010]

Como pequenas mudanças no conjunto de treinamento podem levar a árvores de classificação completamente diferentes, o classificador Random Forests realiza uma média da predição de diferentes árvores de decisão, criadas através de reamostragens do conjunto de treinamento original [Inza et al., 2010].

2.2.1.2 Support Vector Machine

Support Vector Machine (SVM) é uma técnica de aprendizagem supervisionada amplamente utilizada na Bioinformática. Para a criação do modelo, caso as classes sejam linearmente separáveis, o SVM tem como objetivo a definição de um hiperplano de margem máxima(como explicado abaixo). Se, por outro lado, os dados não forem linearmente separáveis, o SVM mapeia as instâncias do conjunto de treinamento em um espaço hiper dimensional, no qual é criado um hiperplano para separar as classes distintas. Para guiar a construção deste hiperplano, outros dois hiperplanos são considerados. Este processo é ilustrado na Figura 2.14.

Um hiperplano (H_1) pode ser considerado como o conjunto de pontos $\mathbf{x}$ que atendam a seguinte equação:

$$\mathbf{w} \cdot \mathbf{x} + b = 0, \tag{2.1}$$

sendo $\mathbf{w}$ um vetor normal ao hiperplano e $\frac{b}{\|\mathbf{w}\|}$ representa o deslocamento do hiperplano ao longo de $\mathbf{w}$ a partir da origem.

Assim, a fim de maximizar a generalização do classificador, é necessário encontrar um hiperplano que maximize a área de separação dos dois hiperplanos paralelos, chamada de margem. A margem de um hiperplano H_1 pode ser compreendida como a distância entre os dois hiperplanos H_{11} e H_{12} [Inza et al., 2010][Cerqueira et al., 2014].

Levando em consideração que $\mathbf{w}$ e b representam um hiperplano H_1, e empregando o algoritmo SVM em uma classe com rótulos -1 e $+1$, uma determinada instância pode ser predita através da seguinte equação [Cerqueira et al., 2014]:

$$\mathrm{y} = \begin{cases} -1, & \text{if } \mathbf{w} \cdot \mathbf{x}_i + b < 0; \\ +1, & \text{if } \mathbf{w} \cdot \mathbf{x}_i + b > 0. \end{cases} \tag{2.2}$$

Ainda de acordo com Cerqueira et al. [2014], $\mathbf{w}$ e b podem ser redimensionados de forma que H_{11} e H_{12} podem ser representados pela seguinte expressão:

$$\begin{aligned} H_{11}: &\quad \mathbf{w} \cdot \mathbf{x} + b = -1, \\ H_{12}: &\quad \mathbf{w} \cdot \mathbf{x} + b = 1. \end{aligned} \tag{2.3}$$

Por meio da geometria, a distância entre H_{11} e H_{12}(a margem da fronteira de decisão) pode ser calculada por $\frac{2}{\|\mathbf{w}\|}$. Então, o objetivo é encontrar $\mathbf{w}$ e b que minimizem $f(\mathbf{w}) = \|\mathbf{w}\|$. Além disto, $\|\mathbf{w}\|$ pode ser substituído por $\frac{1}{2}\|\mathbf{w}\|^2$. Como resultado final, a aprendizagem linear do SVM pode ser representada pelo seguinte problema de

otimização quadrática [Cerqueira et al., 2014]:

$$\begin{aligned} & \min \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{subject to} \quad & y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1, \quad i = 1, 2, ..., N. \end{aligned} \tag{2.4}$$

Na Figura 2.14 é ilustrado o processo de classificação SVM para duas classes. Para classificar uma nova instância, basta localizá-la em um dos lados do hiperplano a partir do vetor de atributos.

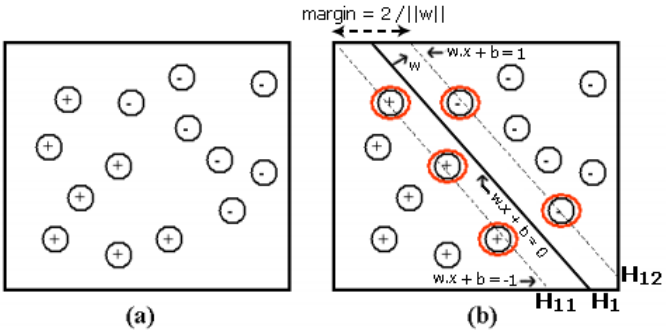

Figura 2.14. A) Conjunto de dados de dois rótulos. B) Modelo SVM. Os dados com círculos vermelhos simbolizam os vetores de suporte. Fonte: adaptado de Tan et al. [2001].

Para resolver este problema de otimização quadrática, podem ser utilizadas técnicas como o método dos multiplicadores de Lagrange juntamente com as condições de Karush-Kuhn-Tucker (KKT). As modificações da formulação matemática acima através destas técnicas e do uso de funções Kernel, resultam nas formulações frequentemente vistas na literatura. A função Kernel tem o papel de mapear os dados não lineares para um novo espaço hiper dimensional através da transformação $\Phi(\mathbf{x})$. Assim, a função Kernel permite que as instâncias no espaço obtido possam ser separadas por uma fronteira de decisão linear [Cerqueira et al., 2014].

Por outro lado, mesmo sendo bastante popular, o método SVM sofre críticas devido a sua complexidade e dificuldade de se verificarem matematicamente os resultados obtidos [Inza et al., 2010].

2.2.1.3 Multilayer Perceptron

As redes neurais artificiais tiveram como inspiração o funcionamento do sistema nervoso dos seres humanos. Os neurônios (células nervosas) transmitem, processam e guardam informações que permitem o funcionamento de sentidos como paladar, olfato, tato, etc. Para realizar estas atividades, o neurônio transmite atividade elétrica através do

axônio. Então, por meio dos terminais dos axônios, podem ser realizadas sinapses para os dendritos de outros neurônios. Além disto, o cérebro humano contém bilhões de neurônios trabalhando em paralelo e distribuídos em rede [Dougherty, 2012].

O modelo matemático perceptron foi desenvolvido visando emular a capacidade humana de reconhecer padrões. Ele foi projetado a partir principalmente do conhecimento sobre a capacidade de processamento paralelo do cérebro humano. Sendo assim, a rede neural artificial utiliza processamento paralelo visando resolver problemas que a computação linear é incapaz de lidar. Inspirada nas características do neurônio, a rede neural artificial utiliza parâmetros que são adaptados ao longo de sua etapa de treinamento visando a melhor forma de solucionar a tarefa empregada [Dougherty, 2012].

A Figura 2.15 ilustra um perceptron. Os nodos de entrada são diretamente ligados aos nodos de saída. Cada valor de entrada é multiplicado a seu peso e os resultados são somados. O valor da soma é comparada a um valor limiar (*Threshold*). Se o valor der acima deste limiar, a saída produzida será 1. Senão, a saída produzida será 0 (é comum também utilizar -1 ao invés de 0).

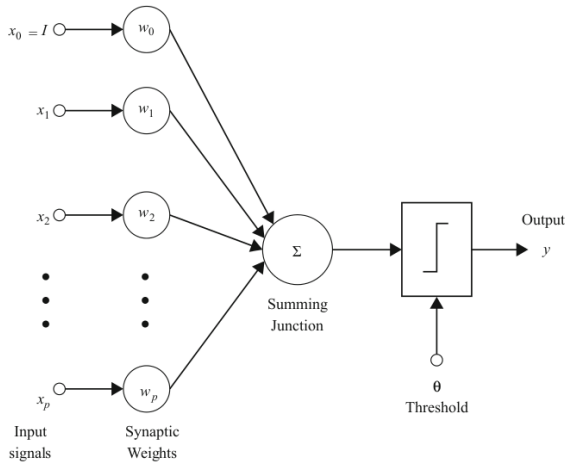

Figura 2.15. Neurônio artificial desenvolvido por McCullough e Pitts (1943)).
Fonte: Dougherty [2012].

Na aprendizagem supervisionada, as entradas e saídas são conhecidas, ou seja, as classes de cada instância são providas. Assim, durante a fase de treinamento de uma rede neural artificial, os pesos são inicializados com valores aleatórios. Em seguida, os pesos são ajustados de tal maneira que a saída do perceptron seja consistente a classe. Para isto, tenta-se encontrar os pesos que minimizem a função objetivo: $E(w) = \frac{1}{2}\sum_{i=1}^{N}(y_i - \tilde{y}_i)^2$.

Algorithm 1 Algoritmo de Aprendizagem Perceptron

procedure PerceptronLearningAlgorithm
Let $D = \{(x_i, y_i | i = 1, 2, ..., N)\}$ be the set of training examples
Initialize the weight vector with random values, w^0
repeat
  for each training example $(x_i, y_i) \epsilon D$
    Compute the predicted output $\tilde{y}_i^k$
    for each weight w_j **do**
      Update the weight, $w_j^{k+1} = w_j^k + \lambda(y_i - \tilde{y}_i^{(k)})x_{ij}$
    end for
  end for
until stopping condition is met
end procedure

O Algoritmo 1 é utilizado para o treinamento. x_{ij} é o valor do j-ésimo atributo do exemplo de treinamento x_i.E λ é a taxa de treinamento utilizada para atenuar o valor da alteração do peso.

Assim, a etapa de treinamento de uma rede neural artificial pode ser compreendida como um problema de otimização. O objetivo é encontrar os pesos que minimizem a soma dos erros quadráticos do conjunto de treinamento.

Após a introdução do perceptron, as redes neurais evoluíram, sendo possível uma arquitetura com várias camadas intermediárias (também chamadas de ocultas, entre a entrada e a saída), onde cada camada pode conter diversos nodos ocultos. Isto permite funções mais sofisticadas entre o vetor de atributos e a saída. Também surgiram novas funções de ativação, ou seja, mapeiam a soma das entradas de uma camada multiplicadas pelos respectivos pesos num valor de saída. A função de ativação mais conhecida é a função logística, que realiza tal mapeamento para valores no intervalo[0,1] que podem ser interpretados como probabilidades[Pang-Ning et al., 2005].

2.3 Trabalhos Relacionados

Os métodos computacionais para descoberta de novos miRNAs / pre-miRNAs podem ser divididos em três categorias: baseados em genômica comparativa, classificadores *ab initio* e métodos completamente ab initio.

A estratégia de genômica comparativa, que busca genes conservados entre espécies com ancestralidade compartilhada é utilizada pelas seguintes ferramentas:

- ERPIN [Lambert et al., 2004] é um servidor web (*http://rna.igmors.u-psud.fr/Software/erpin.php*) que recebe como entrada uma sequência de alinhamento de RNA e então através de programação dinâmica, tenta localizar tRNAs, 5S rRNAs, SRP RNA, C/D box snoRNAs, hammerhead motifs e miRNAs.

- MirAlign [Wang et al., 2005] permite encontrar miRNAs homólogos através da sua ferramenta online (http://bioinfo.au.tsinghua.edu.cn/miralign/). A estratégia utiliza informações da estrutura e de alinhamento da sequência, o que permitiu uma performance superior ao método ERPIN.

- MiRscan [Lim et al., 2003] é um software que visa a identificação de miRNAs em vertebrados. Esta abordagem explora o fato dos miRNAs serem derivados de um *stem loop* precursor filogeneticamente conservado com traços característicos. Sendo assim, através de uma janela de 21 nt, cada *stem loop* (com auxílio do RNAfold [Hofacker, 2003]) é avaliado como possível pre-miRNA, recebendo uma pontuação para então ser comparado com pre-miRNAs validados experimentalmente.

- miRseeker [Lai et al., 2003] com o auxílio do Mfold, tenta localizar sequências de região intergênica conservada que possam formar *stem loop*. Então, seleciona os *stem loops* que os padrões de conservação sejam similares aos do conjunto de referência.loop

- RNAmicro [Hertel & Stadler, 2006] tenta localizar novos miRNAs com uma estratégia de predição da estrutura dos possíveis candidatos com uma análise comparativa de suas sequências. Esta comparação é feita com pre-miRNAs já preditos através de outras ferramentas, como EvoFold e RNAz. Então, estes possíveis miRNAs são classificados utilizando máquinas de vetores de suporte (SVM). O *software* está disponível para download em (http://www.tbi.univie.ac.at/ jana/software/RNAmicro.html).

- BayesMiRNAfind [Yousef et al., 2006] recebe uma sequência genômica, gera estruturas secundárias em uma janela de 110 nt tentando encontrar possíveis stem loops. Então, tenta localizar miRNAs dentro dos candidatos a *stem loops*. Por fim, classifica os possíveis miRNAs através de um classificador Bayesiano chamado Naïve Bayes e aplica um filtro de conservação, visando aumentar a qualidade da busca. É possível realizar o download do *software* em: (http://wotan.wistar.upenn.edu/miRNA).

- MiRFinder [Huang et al., 2007] é um software (http://www.bioinformatics.org/mirfinder/) *Desktop* que utiliza 18 parâmetros da estrutura secundária de possíveis miRNAs e então os compara com os pre-miRNAs conhecidos das espécies relacionadas.

- miRRim [Terai et al., 2007] é um método que utiliza cadeias escondidas de Markov (HMM) para localizar novos miRNAs. São utilizadas características da estrutura secundária e evolucionária para gerar uma sequência de vetores multidimensionais. Foram utilizados 290 miRNAs para treinar os atributos dos vetores.

As seguintes abordagem são utilizadas para classificar, ou seja, como entrada recebem sequências de candidatos à miRNAs ou pre-miRNAs e retornam em geral, verdadeiro ou falso:

- MirBoost ([Tempel et al., 2015]) é um software livre que implementa algoritmos de aprendizagem de máquina bem como o método de *boosting* em conjunto de *weakened* SVM visando a classificação de miRNAs precursores. Inicialmente, a ferramenta recebe como entrada sequências de possíveis pre-miRNAs. Então, para realizar a classificação são extraídos 187 atributos de cada sequência candidata, sendo 32 atributos da haste exata, 45 da haste não exata e 110 da sequência completa. Visando evitar redundância nos atributos, foram utilizadas técnicas como *Greedy Stepwise*, *Scatter Search*, *Best First*, *Linear Forward Selection* e *Subset Size Forward Selection*. Os atributos selecionados por mais de uma técnica são escolhidos. A fim de criar os modelos de classificação, foram utilizados 863 pre-miRNAs humanos contidos na mirBase versão 18, como exemplos positivos. Adicionalmente, são utilizadas regiões exônicas de genes codificantes de proteínas, tRNAs, siRNAs, snRNAs e snoRNAs como exemplos negativos. Além disto, pode-se criar modelos customizados para predição de pre-miRNAs na ferramenta. O resultado final, do método são os candidatos classificados como pre-miRNAs. O software está disponível para download em: evryrna.ibisc.univ-evry.fr/miRBoost/index.html

- miRNA-dis ([Liu et al., 2015]) é um servidor web (http://bioinformatics.hitsz.edu.cn/miRNA-dis/) para classificação de pre-miRNAs de animais, plantas e vírus. Inicialmente o servidor recebe como entrada sequências de candidatos a pre-miRNA em FASTA. Então, para cada sequência informada, são construídos vetores de atributos através da "distância de pares de base". O pacote Vienna [Hofacker, 2003] foi utilizado com o intuito de gerar a estrutura secundária das sequências. Para treinar o algoritmo SVM, foram utilizadas 1872 sequências de pre-miRNAs validados obtidas na mirBase versão 20 como exemplos positivos. Adicionalmente, foram obtidos 8489 pseudo pre-miRNAs [Xue et al., 2005]. A fim de evitar desbalanceamento entre as classes, nenhuma sequência com homologia foi selecionada. Por fim, 1612

sequências foram aleatoriamente escolhidas como exemplos negativos. Como resultado final, cada sequência candidata é rotulada como real ou falsa. Para avaliar o método, foram utilizadas sequências de pre-miRNAs animais, plantas e vírus obtidas na mirBase versão 21. Para evitar overfitting, as sequências com mais de 80 % de homologia com as sequências humanas foram retiradas. O método apresentou uma média de 87.02 % de precisão em outras espécies.

- HuntMi ([Gudyś et al., 2013]) é um software de classificação de pre-miRNAs. Como entrada, recebe candidatos a pre-miRNAs em formato FASTA. Como saída, informa se cada sequência enviada é verdadeiramente um pre-miRNA. A ferramenta conta com modelos prontos para classificar pre-miRNAs de plantas, animais, vírus, *H. sapiens* e *A. thaliana*. Além disto, é possível criar novos modelos *datasets* customizados pelo próprio usuário. A fim de criar os modelos originais, os exemplos positivos foram obtidos a partir de todos os pre-miRNAs validados experimentalmente contidos da mirBase versão 17. Adicionalmente, foram utilizados RNAs mensageiros(mRNAs) de animais, plantas e vírus como exemplos negativos. Então, foram criadas subsequências dos mRNAs, o início da sequência foi aleatoriamente escolhido e o final calculado para que os exemplos negativos tivessem o mesmo tamanho dos positivos, garantindo assim a mesma distribuição de tamanhos nas duas classes. Para realizar a classificação, além dos 21 atributos utilizados no software microPred, foram criados sete novos atributos. A fim de resolver o desbalanceamento entre a quantidade de exemplos positivos e negativos, foram testadas diversas estratégias e classificadores como: Naive Bayes, Multilayer Perceptron, Support Vector Machine, Random Forest, APLSC, SMOTE e ROC Select. Como resultado final, Random Forest + ROC Select apresentaram os melhores resultados. Em consequência disto, foram encapsulados em um *plugin*, o que possibilita incorporá-lo em outras aplicações. O software HuntMi, datasets e plugin estão disponíveis para download em http://adaa.polsl.pl/agudys/huntmi/huntmi.h213002tm.

- HeteroMirPred ([Lertampaiporn et al., 2013]) é um *webserver* que realiza a classificação de miRNas precursores. Inicialmente, recebe sequências de possíveis pre-miRNAs. Então, são extraídos 125 atributos de cada sequência informada. Assim, os dados são classificados por um conjunto de algoritmos: KNN, Random Forest e SVM. Para treinar os classificadores, foram utilizados 600 pre-miRNAs de *Homo sapiens*, 200 de *Oryza sativa* e 200 de *Arabidopsis thaliana*, obtidos da mirBase 17, como exemplos positivos. Adicionalmente, foram utilizadas sequências de 4000 pseudo pre-miRNAs [Xue et al., 2005], extraídos de regiões codificantes

do genoma humano, como exemplos negativos. A fim de resolver o problema de desbalanceamento entre as classes, a técnica SMOTE foi aplicada nas instâncias positivas, para aumentar em 50 %o número de exemplos. Por outro lado, é feito uma re-amostragem dos exemplos negativos, retirando 25 % das instâncias. Então, são gerados quatro modelos de cada algoritmo a partir dos *datasets* balanceados. Por fim, é realizada uma seleção dos melhores atributos. Como resultado final, são apresentados como verdadeiros pre-miRNAs as sequências candidatas aprovadas pelo conjunto de classificadores.

- PlantMiRNAPred ([Xuan et al., 2011]) é um servidor web (http://nclab.hit.edu.cn/PlantMiRNAPred/) para classificação de pre-miRNAS de plantas. Inicialmente, recebe como entrada sequências de candidatos a pre-miRNA em formato FASTA. Então, extrai 115 atributos das estruturas primárias e secundárias - geradas com o auxílio do software RNAfold [Hofacker, 2003] a partir da sequência. Posteriormente, estes dados são enviados ao algoritmo SVM. Visando maximizar a eficiência da classificação, os atributos redundantes são descartados. Por fim, os atributos são ordenados pelo ganho de informação. Aqueles que atendam os pontos de corte mínimos são selecionados. Para realizar o treinamento do classificador, foram utilizados 1906 pre-miRNAs de plantas obtidos na mirBase versão 14, como exemplos positivos. Adicionalmente, foram utilizadas 2122 sequências das espécies A.*thaliana* e G.*max*, extraídas das regiões que codificam proteínas como exemplos negativos. Como resultado final, os candidatos são classificados como pre-miRNAs reais ou pseudo pre-miRNAs.

- microPred ([Batuwita & Palade, 2009]) é um software que utiliza técnicas de aprendizagem de máquina para classificar pre-miRNAs. Inicialmente, recebe como entrada sequências de possíveis pre-miRNAs em formato FASTA. Então, cada uma das sequências candidatas tem sua estrutura secundária gerada utilizando o RNAfold Hofacker [2003] e 49 atributos são extraídos. Após a extração, são selecionados os 21 melhores atributos. Por fim, os dados são enviados ao algoritmo SVM. Para realizar o treinamento do classificador, como exemplos positivos foram utilizadas 691 sequências não redundantes de pre-miRNAs humanos, obtidas através da mirBase 12. Adicionalmente, foram utilizados 8494 pseudo pre-miRNAs humanos [Xue et al., 2005] e 754 sequências de ncRNAs (exceto miRNAs) como exemplos negativos. No resultado final, são mostrados os candidatos classificados como pre-miRNAs. O software está disponível para download em http://www.cs.ox.ac.uk/people/manohara.rukshan.batuwita/microPred.htm

- MiPred ([Jiang et al., 2007]) é um servidor web (http://www.bioinf.seloopu.edu.cn/miRNA/) que realiza a classificação de miRNA precursores. Como entrada, recebe sequências de possíveis pre-miRNAs em formatos FASTA, CGC, GeneBank ou EMBL. Inicialmente, em cada sequência informada é avaliada se sua estrutura secundária forma um *hairpin*. Caso positivo, são extraídas características da sequência e os dados são enviados ao algoritmo *Random Forest*. Para realizar o treinamento do classificador, foram obtidos 462 pre-miRNAs humanos na mirBase versão 8.2. Então, as sequências com múltiplos *loops* foram descartadas. Assim, 426 pre-miRNAs humanos foram utilizados como exemplos positivos. Adicionalmente, foram obtidos 8494 pseudo pre-miRNAs humanos [Xue et al., 2005] e descartadas as sequências que não atenderam aos seguintes critérios: tamanhos mínimo e máximo, quantidade de pareamentos, energia livre e ausência de múltiplos *loops*. Por fim, as sequências restantes foram utilizadas como exemplos negativos. Em seu resultado final, as sequências informadas são rotuladas como pre-miRNAs reais ou falsos.

- Triplet-SVM (Xue et al. [2005]) É um software para classificação de pre-miRNAs. Inicialmente, recebe como entrada sequências de candidatos a pre-miRNA em formato FASTA. Então, utiliza o software RNAfold [Hofacker, 2003] para gerar a estrutura secundária de cada sequência informada. Assim, cada estrutura secundária tem seus *triplets* extraídos, contabilizados e normalizados. Por fim, os valores encontrados são enviados ao algoritmo SVM. Para realizar o treinamento do classificador, foram utilizados 193 pre-miRNAs humanos que não apresentam múltiplos *loops*, obtidos mirBase 5, como exemplos positivos. Adicionalmente, foram utilizadas 8494 sequências que formam *hairpins*, obtidas de regiões codificantes do DNA humano, como exemplos negativos. Como resultado final, os candidatos classificados pelo algoritmo SVM para definir os candidatos mais prováveis. O software está disponível para download em: http://bioinfo.au.tsinghua.edu.cn/mirnasvm/.

Por fim, são exemplos de abordagens completamente ab initio para predição de miRNAs ou pre-miRNAs, ou seja, recebem uma grande sequência genômica e retornam possíveis novos genes:

- MirnaSearch ([Titov & Vorozheykin, 2013]) utiliza Modelos Ocultos de Markov (HMM), treinadas com 1872 pre-miRNAs humanos obtidos da mirBase (versão 20) como exemplos positivos e 1872 sequências aleatórias de pseudo-precursores como *dataset* negativo. Para realizar a busca por novos ge-

nes de pre-miRNAs, utiliza os padrões das estruturas primárias e secundárias (gerada através do *software* GaRNA), dos pre-miRNAs em uma determinada sequência genômica informada. Após um pre-miRNA ser predito, é realizada outra análise para encontrar o miRNA maduro contido no mesmo. Como resultado final, são mostrados pre-miRNA, miRNA maduro e a estrutura secundária de ambos. Este método está disponível em um servidor web (*http://wwwmgs.bionet.nsc.ru/mgs/programs/rnaanalys/index.html*). A HMM foi treinada apenas em genes de *Homo sapiens*, em consequência disto, embora seja possível realizar a busca utilizando o genoma de qualquer espécie, não são esperados os mesmos resultados de seletividade e sensibilidade.

- miRNAFold ([Tempel & Tahi, 2012]) é um servidor web para localizar pre-miRNAs (http://evryrna.ibisc.univ-evry.fr/miRNAFold/). Inicialmente, são geradas janelas a partir de uma sequência genômica, parâmetros da espécie, porcentagem de atributos a serem verificados e tamanho da janela informados. Em cada janela, é construída uma matriz de pares de base para a respectiva subsequência e são realizadas buscas por hastes longas exatas. Então, para cada haste longa encontrada são extraídos 12 atributos. As hastes exatas mais prováveis sofrem uma extensão, permitindo encontrar hastes longas não exatas. Assim, cada haste longa não exata tem 18 atributos extraídos. Em consequência disto, as hastes longas não exatas mais prováveis são estendidas visando reconstituir a estrutura completa dos pre-miRNAs candidatos. A análise de cada etapa é realizada a partir dos valores observados ao estudar os atributos dos pre-miRNAs de *Homo sapiens*, *Mus musculus*, *Rattus norvegicus*, *Gallus gallus*, *Xenopus tropicalis*, *Danio rerio*, *Drosophila melanogaster*, *Bombyx muri*, *Caenorhabditis elegans*, *Ciona intestinalis*, *Zea mays*, *Medicago truncatula*, *Oryza sativa* e A*rabidopsis thaliana*-*contidos* contidos na mirBase versão 17. Por fim, os pre-miRNAs que atendam os requisitos esperados são mostrados.

- miRPara ([Wu et al., 2011]) é um software livre para predição de miRNAs maduros e pre-miRNAs a partir de grandes sequências genômicas. Inicialmente, a sequência de DNA informada é dividida em fragmentos de 500 nt com 200 nt de sobreposição. Para cada fragmento de 500 nt, é utilizado o software UNAFold [Markham & Zuker, 2008] para predição de candidatos a pre-miRNAs. Então, são rejeitados os *hairpins* com menos de 60 nt e aquelas que contenham segmentos com sobreposição. Em consequência disto, os demais *hairpins* são reavaliados com o UNAFold. Após esta etapa, os *hairpins* restantes são investigados visando encontrar candidatos a miRNA. Para realizar esta análise, são extraídos

77 parâmetros de cada candidato. Então, é realizada uma classificação utilizando o algoritmo SVM. Para a criação dos modelos de classificação do SVM, foram obtidas, no mirBase versão 13, 5576 sequências validadas de miRNAs / pre-miRNAs utilizadas como exemplos positivos. Então, foram criados três modelos: animal, plantas e global, o primeiro com 4886 sequências de animais, o segundo com 1215 de plantas e terceiro com todas as 5576 sequências validadas. Adicionalmente, foram criadas 5576 sequências artificiais para construir o *datasets* negativos. No intuito de maximizar a performance, em cada modelo foram adicionados exemplos negativos da ordem de 1:1 até 1:20. Por fim, aqueles candidatos a pre-miRNA / miRNA que são classificados como verdadeiros pelo SVM são mostrados como resultado final. O software está disponível para download em https://code.google.com/p/mirpara/.

- CID-miRNA ([Tyagi et al., 2008]) é uma ferramenta para predizer genes de pre-miRNAs a partir de uma sequência de DNA, Score Cutoff, Structural Score Cutoff, tamanhos mínimos da haste terminal e janela informados em seu *webserver* (http://mirna.jnu.ac.in/cidmirna/). Para realizar a predição, foi construído modelo de *Stochastic Context Free Grammar* (SCFG). A fim de construir a gramática, foram utilizadas características das estruturas primárias e secundárias de 474 sequências de pre-miRNAs humanos validados como exemplos positivos (obtidos na mirBase). Para garantir que não houvesse duplicidades, as sequências com alta homologia foram retiradas. Adicionalmente, foram utilizadas 300 sequências de RNA ribossomal humano (rRNA) como exemplos negativos (obtidas no GenBank). Então, o algoritmo de CYK foi utilizado para definir um *Score* através da triagem dos conjuntos positivos e negativos. Por fim, foi utilizado um classificador J48 através do Weka toolkit para construir uma árvore de classificação visando determinar o melhor *Cut off* entre os *datasets* positivos e negativos. Como resultado final, são apresentados os pre-miRNAs que atendam aos *Scores* definidos assim como suas respectivas estruturas secundárias.

- Vmir ([Grundhoff et al., 2006]) é um programa que realiza a predição de novos pre-miRNAs em eucariotos. Como entrada, recebe valor mínimo do *Score*, sequência a ser analisada, tamanho da janela e do passo, tamanhos mínimo e máximo do *hairpin*. Inicialmente, utiliza uma janela de tamanho informado e avança nesta em passos de tamanho definido. Então, utiliza o RNAfold [Hofacker, 2003] para gerar as estruturas secundárias dos *hairpins* contidos na janela. Aqueles candidatos que atendam ao tamanho mínimo informado são selecionados e analisados. Por fim, cada candidato tem suas características comparadas às sequências de

um *dataset* com 175 pre-miRNAs humanos conhecidos e 5500 pseudo *hairpins*. Características como porcentagem de nucleotídeos pareados, tamanho da maior haste, posição relativa do *loop* terminal e ocorrência de bojos são utilizadas. Os pre-miRNAs que atinjam uma pontuação 0 ou 1 são descartados. Já os candidatos com pontuação 2 são mostrados. O software está disponível para download em: http://booksite.elsevier.com/9780123739179/.

Capítulo 3

Materiais e Métodos

Neste capítulo será mostrado o método *ab initio* de predição de novos pre-miRNAs em *Homo sapiens* e *Mus musculus* proposto neste trabalho.

O objetivo principal da nossa abordagem é predizer novos pre-miRNAs a partir de uma sequência de DNA, sem utilizar outras informações biológicas (método completamente ab initio). A predição de novos pre-miRNAs é realizada em três etapas. Na primeira é a efetuada uma busca por hastes exatas, na segunda uma busca por hastes longas não exatas e na terceira uma busca pelo *hairpin* completo. Para filtrar os possíveis pre-miRNAs, em cada etapa é feita uma classificação, o que requer três modelos de aprendizagem de máquina diferentes.

3.1 Características dos pre-miRNAs

Visando encontrar atributos das estruturas secundárias dos pre-miRNAs, Tempel & Tahi [2012] realizaram o download e estudaram 16.772 pre-miRNAs disponíveis na mirBase, *release* 17. Eles observaram que quase sempre os pre-miRNAs eram compostos de uma haste longa exata. A haste longa exata é uma subsequência (*p, p'*) que apresenta uma sucessão de pares de base A-U, GU e C-G e pode ser definida como:

$$(i) \quad |p| = |p'| = m$$
$$(ii) \quad p[k] \; R_c \; p'[m-k+1], \quad \forall k, 1 \leq k \leq m$$

onde (R_c) é a relação de complementaridade entre os nucleotídeos A R_cU, G R_c U e G R_c C. Acompanhando a Figura 3.1, pode-se verificar que todos os pre-miRNAs

contidos no mirBase continham pelo menos uma haste exata de tamanho 5 nt ou superior.

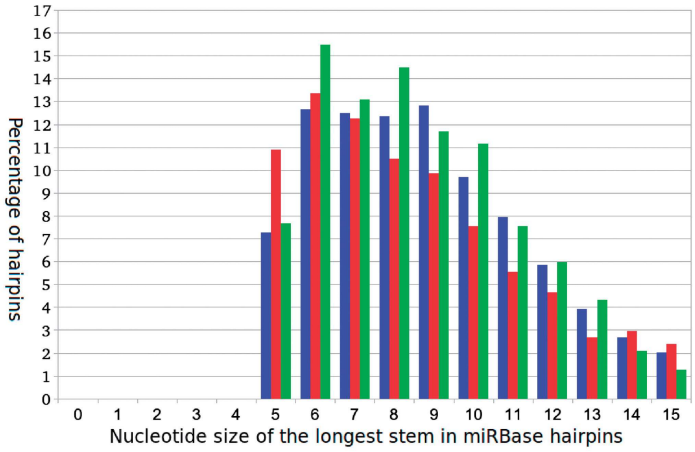

Figura 3.1. Tamanho das hastes exatas dos pre-miRNAs. Todos os pre-miRNAs contém uma haste longa exata. *Homo sapiens* em vermelho, *Mus musculus* em verde e todas as espécies em azul. Fonte: Tempel & Tahi [2012]

Por consequência disto, é utilizada neste trabalho a estratégia de localizar as hastes exatas dos pre-miRNAs em uma determinada sequência de DNA, para então, tentar aproximar a estrutura do pre-miRNA que a contém.

Outra constatação feita por Tempel & Tahi [2012] foi que quase todos os pre-miRNAs contidos no mirBase continham uma haste longa não exata. A haste longa não exata é uma sucessão de hastes exatas separadas por *loops* simétricos, os quais são menores do que as hastes exatas que os cercam. Sendo assim, as hastes longas não exatas podem ser definidas como:

$$(i) \quad |p| = |p'| = m$$
$$(ii) \quad p[k] \ \overline{R_{nc}} \ p'[m-k+1], \quad \forall k, 1 \leq k \leq m$$

onde $\overline{R_{nc}}$ representa a relação de não complementaridade entre os nucleotídeos. O tamanho do *loop* simétrico é mensurado através do número de nucleotídeos que não formam pares de base em um lado do *loop*. Na Figura 3.2 é possível constatar que as hastes longas não exatas constituem boa parte dos pre-miRNAs.

Sendo assim, também é utilizada no presente trabalho a estratégia de após localizar as hastes exatas dos pre-miRNAs, estender a busca tentando localizar a haste longa não exata que faz parte do pre-miRNA, visando por fim, encontrar a estrutura completa do mesmo.

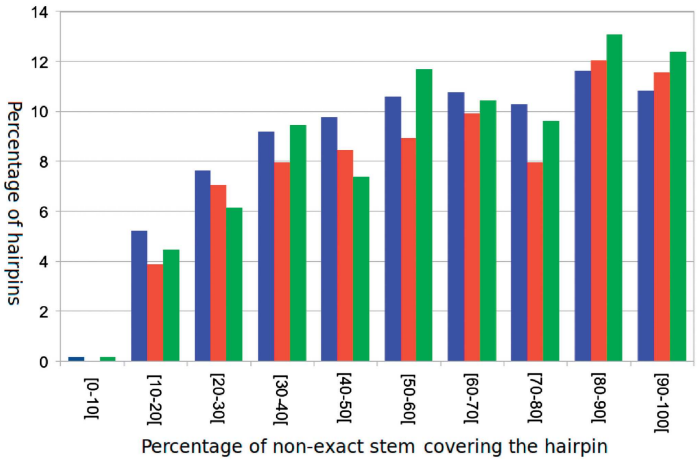

Figura 3.2. Porcentagem de pre-miRNAs que possuem haste não exata. Praticamente todos os pre-miRNAs contém uma haste longa não exata. *Homo sapiens* em vermelho, *Mus musculus* em verde e todas as espécies em azul. Fonte: Tempel & Tahi [2012]

3.2 Conjuntos de Treinamento

Os modelos de aprendizagem de máquina foram construídos a partir de conjuntos de treinamento cujos exemplos positivos foram obtidos a partir de sequências de pre-miRNAs presentes na miRBase release 21 [Griffiths-Jones et al., 2006] (http://www.mirbase.org/).

Visando minimizar a adição de ruído nos conjuntos de treinamento, foram utilizadas apenas sequências validadas experimentalmente. Nos conjuntos de treinamento da sequência artificial humana, humana genômica e rato genômica foram utilizados 236, 295, 364 pre-miRNAs respectivamente. Além disto, para evitar *overfitting*, os pre-miRNAs contidos nos *test sets* descritos abaixo foram retirados dos *training sets*. Os *test sets* serão mais bem detalhados no próximo capítulo.

Adicionalmente, o conjunto de treinamento com exemplos negativos para humano e rato consistiu de 2480 e 3298 genes de snRNAs, snoRNAs, tRNAs e miscRNAs os quais foram obtidos no genbank e NRDR database [Paschoal et al., 2012] (http://bioinfo-tool.cp.utfpr.edu.br/nrdr/). Adicionalmente, foi utilizado um *dataset* composto de 1872 pseudo pre-miRNAs [Xue et al., 2005] no treinamento dos modelos de Homo sapiens, totalizando 4254 exemplos negativos.

A Figura 3.3 mostra como foi realizado o treinamento de aprendizagem de máquina na presente abordagem. Para cada sequência informada, o software Vienna [Hofacker, 2003] foi utilizado para gerar a estrutura secundária, depois são extraídos os atributos e, por fim, são gerados três modelos. Um modelo para a primeira etapa da

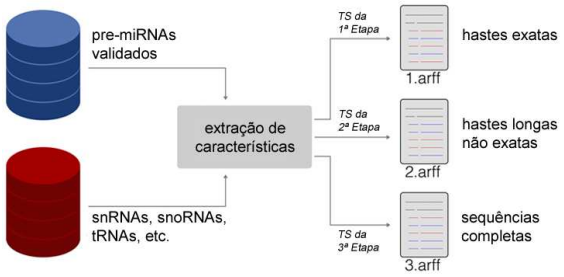

Figura 3.3. Pre-miRNAs foram utilizados como exemplos positivos. Por outro lado, para exemplos negativos foram utilizados snRNAs, snoRNAs, tRNAs, miscRNAs e sequências artificiais.

busca, que realiza a procura por hastes exatas. Outro modelo para a segunda etapa da busca, que realiza procura por hastes longas não exatas. E, por fim, o terceiro modelo para a classificação final da sequência completa. Este método está encapsulado em um servidor web, pois através deste, qualquer usuário registrado pode realizar *upload* de sequências de genes em arquivos FASTA a fim de criar seu conjunto de treinamento customizado.

3.3 Aprendizagem Desbalanceada

Como pode ser visto na descrição dos nossos dados, os conjuntos de treinamento são altamente desequilibrados. Ou seja, o número de exemplos negativos é muito maior do que o número de exemplos positivos. Isto pode ser um problema, pois o modelo resultante pode ser tendencioso a classe dominante, apresentando uma precisão ineficiente para classificar exemplos positivos.

Então, a fim de solucionar o problema de desbalanceamento entre a proporção de classes positivas e negativas, foram testadas três estratégias: matriz de custos [Hall et al., 2009], amostragem de *dataset* [Hall et al., 2009] e o filtro SMOTE [Chawla et al., 2011] junto dos algoritmos Random Forest (RF), Support Vector Machine (SVM), Multilayer Perceptron (MLP) e Sequential Minimal Optimization (SMO). Este processo é ilustrado na Figura 3.4.

Em matrizes de custo, é possível definir o custo em caso de erro na classificação das instâncias. O custo de um erro de classificação de uma instâncias positiva, que pertence à classe minoritária, é muito maior do que o custo de um erro de classificação de uma instância negativa. Como resultado, os pesos de ambas as classes na etapa de treinamento são equalizadas.

As abordagens de amostragem e SMOTE alteram o número de instâncias do con-

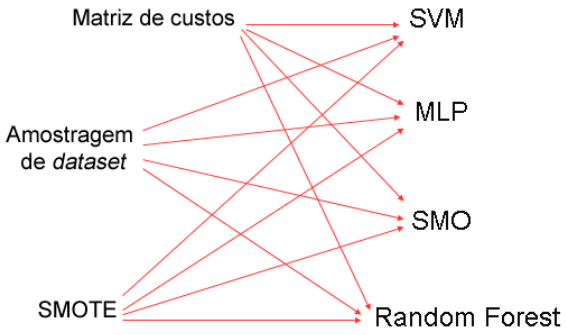

Figura 3.4. Estratégias de desbalanceamento das classes positivas e negativas. Para cada um dos algoritmos de aprendizagem de máquina, foram utilizadas as técnicas de matriz de Custos, reamostragem e SMOTE.

junto de treinamento de forma que ambas as classes contenham o mesmo número de exemplos. Na amostragem, a classe com maior quantidade de exemplos é reduzida, enquanto a classe com menor quantidade não é alterada. Por outro lado, a técnica SMOTE cria instâncias sintéticas para a classe com menor número de exemplos, baseadas nas características dos exemplos existentes combinadas com seus vizinhos mais próximos. Ou seja, ao contrário da técnica de amostragem, não há perda de informação.

Como métricas para avaliação dos modelos gerados através destas estratégias, foram utilizadas a sensibilidade, a seletividade e a média geométrica de ambas, pois esta é comumente utilizada em problemas de aprendizagem desbalanceada em miRNAs [Gudyś et al., 2013]. Além disto, foram realizados treinamentos em cada um dos *datasets* através de validação cruzada(10-fold) para decidir quais técnicas de AM deveriam ser incorporadas na ferramenta computacional. Os resultados destas experiências são apresentados nas Tabelas 3.1, 3.2 e 3.3.

Como mostrado na Tabela 3.1, a utilização do filtro SMOTE se mostrou a técnica mais eficiente para o desbalanceamento em todas as etapas. Quanto aos algoritmos, com uma média geométrica de 97,23% frente a 93,36% do SVM e 91,10% do MLP no *dataset* da primeira etapa, o algoritmo Random Forest apresentou resultados superiores às demais abordagens. Além disto, na segunda etapa Random Forest obteve uma média geométrica de 98,72% frente a 97,52% do MLP e 93,13% do SMO. Por fim, na terceira etapa, o MLP com 99,80% obteve um resultado ligeiramente superior ao Random Forest (99,65%) e SMO (99,60%).

Além disto, como mostrado na Tabela 3.2, o algoritmo Random Forest se mostrou o mais eficiente em todas as etapas do *dataset Homo sapiens*. Na primeira etapa, obteve uma média geométrica de 97,75 %, frente a 93,28 % do algoritmo SVM. Já na segunda etapa, obteve 98,80 % contra 97,45 % do MLP. Finalmente, obteve 99,82 % na terceira

Tabela 3.1. Comparação entre diferentes métodos de desbalanceamento no *dataset* Humano Artificial. Os resultados de sensibilidade (Sen.) e seletividade (Sel.) foram obtidos utilizando *10-fold cross-validation* nos *datasets* de cada etapa do algoritmo.

Método	1a etapa			2a etapa			3a etapa		
	Sen.	Sel.	M.G.	Sen.	Sel.	M.G.	Sen.	Sel.	M.G.
				LibSVM					
Cost matrix	4,7	64,7	17,4	3,8	81,8	17,7	4,2	100	20,6
Sampling	99,6	55,4	74,3	99,6	54,8	73,8	100	65	80,6
SMOTE	87,5	99,6	93,4	82,5	100	90,8	97,9	100	98,9
				SMO					
Cost matrix	80,9	17,9	38	85,6	40	58,1	97,9	77,5	87,1
Sampling	84,3	77,7	81	86	90,6	88,3	98,7	96,7	97,7
SMOTE	86,3	83,5	84,9	91,9	94,4	93,1	99,9	99,3	99,6
				MLP					
Cost matrix	74,1	16,7	35,2	79,2	49,7	62,7	98,7	3,9	19,6
Sampling	78,8	77,5	78,2	89,8	88,3	89,1	97	97,9	97,4
SMOTE	91,5	90,8	91,1	98	97,1	97,5	99,9	99,7	99,8
				RF					
Cost matrix	46,2	44	45,1	74,6	76,5	75,5	89	89	89
Sampling	84,3	78,7	81,4	87,7	87,7	87,7	97,9	97,1	97,5
SMOTE	98,2	96,3	97,2	99,1	98,4	98,7	99,9	99,4	99,7

etapa frente a 99,81 % do MLP e 99,61 do SMO.

Por fim, como mostrado na Tabela 3.3, com uma média geométrica de 95,50% frente a 93,96% do SVM e 87,81% do MLP no *dataset* da primeira etapa, o algoritmo Random Forest apresentou resultados superiores as demais abordagens no *dataset Mus musculus*. Além disto, na segunda etapa Random Forest obteve uma média geométrica de 97,84% frente a 95.05% do MLP e 92,00% do SMO. Por fim, na terceira etapa o MLP com 99,69% obteve um resultado ligeiramente superior ao Random Forest (99,67%) e SMO (99,07%).

Como resultado final, o algoritmo Random Forest, junto do filtro SMOTE, foi definido como o algoritmo em todas as etapas do algoritmo de predição. Muito embora o MLP tenha obtido uma ligeira superioridade (0,15 %) ao Random Forest na terceira etapa do *dataset* artificial humano e *Mus musculus*, ele não foi utilizado devido ao seu tempo de treinamento ser bastante oneroso frente ao algoritmo Random Forest e porque nas demais etapas, o algoritmo RF foi superior.

Tabela 3.2. Algoritmos com filtro SMOTE no *dataset Homo sapiens*. Resultados de sensibilidade e seletividade utilizando 10-fold cross-validation nos *datasets* referentes a cada etapa do algoritmo. Os melhores resultados estão destacados em negrito.

Algoritmo	1 etapa			2 etapa			3 etapa		
	Sen.	Sel.	M.G.	Sen.	Sel.	M.G.	Sen.	Sel.	M.G.
SVM	87,50	99,45	93,28	84,65	100	92,00	98,12	100	99,05
SMO	86,67	83,97	85,27	92,89	94,52	93,70	99,88	99,35	99,61
MLP	92,27	88,39	90,30	97,93	96,99	97,45	99,93	**99,70**	99,81
RF	**97,81**	**97,70**	**97,75**	**98,66**	**98,96**	**98,80**	**99,96**	99,69	**99,82**

Tabela 3.3. Algoritmos com filtro SMOTE no *dataset Mus musculus*. Resultados de sensibilidade e seletividade utilizando *10-fold cross-validation* nos *datasets* referentes a cada etapa do algoritmo. Os melhores resultados estão destacados em negrito.

Algoritmo	1 etapa			2 etapa			3 etapa		
	Sen.	Sel.	M.G.	Sen.	Sel.	M.G.	Sen.	Sel.	M.G.
SVM	89,38	98.79	93,69	100	60,49	77,77	97,61	100	98,79
SMO	85,56	81,57	83,54	91,96	92,46	91,70	99,17	98,98	99,07
MLP	88,89	86,76	87,81	95,20	94,91	95,05	**99,89**	99,51	**99,69**
RF	**96,39**	**94,63**	**95,50**	**97,81**	**97,87**	**97,83**	99,79	**99,57**	99,67

3.3.1 Atributos

Cada etapa de busca tem seu próprio *dataset* assim como seus próprios atributos. Na etapa de localização das hastes exatas, são extraídos os seguintes atributos:

- Tamanho da haste exata
- Delta G
- Porcentagem de A, C, G e U
- Número de A, C, G e U consecutivos
- Porcentagem de Pareamento GU.
- Diferença entre G+A.
- Diferença entre C+U.
- Dinueleotídeos.

A fim de entender o ganho de informação, ou seja, o quão eficiente é um determinado atributo para a separação das classes, a Tabela 3.4 mostra o resultado da execução do algoritmo *InfoGainAttributeEval* Hall et al. [2009] aplicado no conjunto de treinamento da primeira fase.

Tabela 3.4. Ganho de informação dos atributos da primeira fase no conjunto de treinamento da sequência artificial humana .

Atributo	Ganho de informação
delta G	0.08898
size	0.07178
gu-pairing	0.05776
gc-dif-gc	0.05776
di-ag	0.04908
di-gu	0.04821
di-uu	0.03977
di-ac	0.03975
gc-c	0.03915
gc-u	0.03915
di-cc	0.03900
di-uc	0.03660
gc-dif-gagu	0.03256
di-ca	0.02882
di-ug	0.02819
di-cg	0.02647
consecutive-c	0.02530
di-cu	0.02467
di-ga	0.02287
gc-gc	0.01150
di-au	0.00939
gc-au	0.00934
di-gc	0.00932
consecutive-a	0.00515
consecutive-g	0.00402
di-gg	0.00302
consecutive-u	0.00199

A segunda etapa, que realiza uma extensão na intenção de localizar hastes longas não exatas, utiliza os seguintes atributos:

- Tamanho da haste longa não exata
- Delta G
- Porcentagem de A, C, G e U

- Número de A, C, G e U consecutivos
- Tamanho médio da Palíndrome
- Porcentagem de pareamento GU.
- Porcentagem de pareamento de pares de base.
- Diferença entre G+A.
- Diferença entre C+U.
- Porcentagem da diferença G e C.
- Número de loops simétricos
- Tamanho médio dos loops simétricos
- Tamanho do maior loop simétrico
- Quantidade de loops simétricos
- Dinucleotídeos.

Novamente, a fim de entender o ganho de informação de cada um dos atributos, a tabela 3.5 mostra o resultado da execução do algoritmo *InfoGainAttributeEval* aplicado no conjunto de treinamento da segunda fase.

Por fim, a classificação dos candidatos a pre-miRNA é realizada através da extração dos seguintes atributos:

- Tamanho do *hairpin*
- Delta G
- MFE 1 , MFE 2
- Porcentagem de A, C, G e U
- Número de A, C, G e U consecutivos
- Porcentagem de pareamento GU.
- Porcentagem de pareamento GC.
- Porcentagem de pareamento de pares de base.

Tabela 3.5. Ganho de informação dos atributos da segunda fase no conjunto de treinamento da sequência artificial humana .

Atributo	Ganho de informação
delta G	0.16683
size	0.14152
gu-pairing	0.09808
gc-dif-gc	0.08459
number-of-exact-stems-two	0.08063
num-loops	0.06945
di-ga	0.06691
di-ac	0.06641
di-gu	0.06205
di-cc	0.06125
di-au	0.06124
consecutive-c	0.05899
di-ua	0.05882
di-cu	0.05875
gc-c	0.05849
di-gc	0.05671
di-uc	0.05635
gc-dif-gagu	0.05608
di-gg	0.05487
di-ug	0.05208
di-ag	0.04943
di-ca	0.04744
di-uu	0.04563
di-cg	0.04557
max-loops	0.04174
gc-u	0.04153
[a] gc-au	0.03870
media-loops	0.03829
di-aa	0.03667
gc-gc	0.03175
consecutive-g	0.01923
consecutive-a	0.01539
palindrome	0.01418
consecutive-u	0.00208

- Diferença entre G+A.
- Diferença entre C+U.
- Porcentagem da diferença G e C.

- Porcentagem de pareamento nos Hairpins.
- Tamanho do loop simétrico interno.
- Porcentagem de hastes longas não exatas.
- Tamanho da maior diferença entre os dois lados do bojo.
- Tamanho do maior bojo de cada lado.
- Número de bojos consecutivos.
- Número de bojos consecutivos do mesmo lado.
- Dinucleotídeos.
- Triplets.

3.4 Predição ab initio

Para realizar a predição de novos pre-miRNAs, é necessário informar a sequência de DNA que será investigada (Figure 3.7 A), o modelo (gerado durante o treinamento), tamanho mínimo da haste exata, tamanho da janela deslizante, ponto de corte de cada etapa, tamanhos mínimo e máximo dos pre-miRNAs. A Figura 3.5 mostra a interface de Mirnacle *Desktop* para de predição de novos miRNAs.

Então é gerada uma matriz triangular de pares de base nxn, onde n é o tamanho da subsequência sendo analisada. Para construir a matriz M, é utilizado o seguinte Algoritmo 2: A primeira etapa do algoritmo busca por hastes exatas na matriz de pares de base (Figura 3.7 B). Uma haste exata é composta de pares de base AU, GC ou GU (Figura 3.6 A). Em todas as hastes exatas encontradas (de tamanho igual ou superior ao parâmetro informado) são realizadas extrações de atributos. As hastes com classificação igual ou superior ao ponto de corte informado são selecionadas (Figura 3.7 C).

Na segunda etapa, a partir da posição das hastes exatas selecionadas previamente, é realizada uma extensão nas mesmas procurando por hastes longas não exatas nas diagonais esquerda e direita da matriz (Figura 3.7 D). Uma haste longa não exata é a junção de hastes exatas entre *loops* simétricos, de forma que cada *loop* simétrico seja menor do que as hastes conectadas a ele (Figura 3.6 B). Em cada uma das hastes longas não exatas, são realizadas extrações dos atributos. Novamente, as instâncias são classificadas (Figura 3.7 E). As hastes longas não exatas com classificação igual ou superior ao ponto de corte informado são selecionadas.

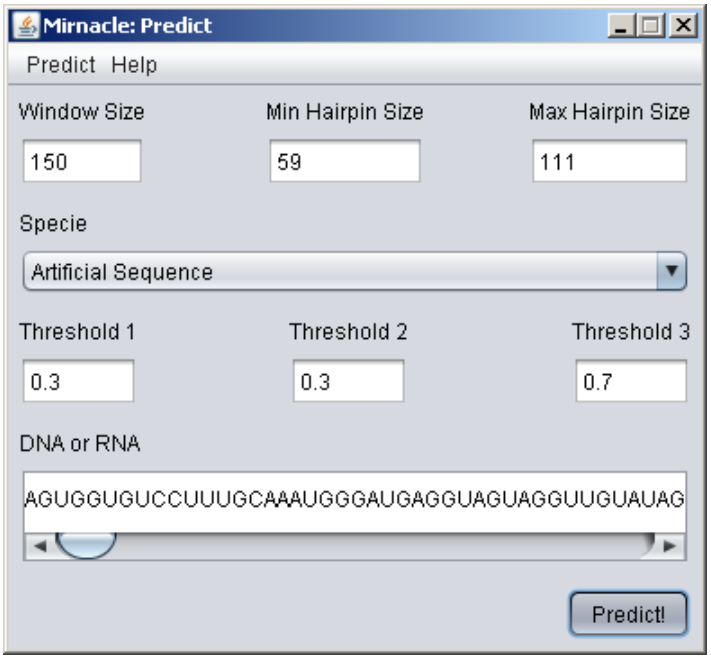

Figura 3.5. Mirnacle *Desktop*: Interface de predição

Algorithm 2 Construção de uma matriz triangular de pares de base

```
1: procedure BasePairingMatrix(s[0..n-1])
2: input: A putative pre-miRNA sequence of length n.
3: output: A triangular base pairing matrix.
4:     for i ← 0 to n-2 do
5:         if s[0] complements s[n-1-i] then
6:             M[i,0] ← 1
7:         else
8:             M[i,0] ← 0
9:         end if
10:        if s[n-1] complements s[i] then
11:            M[0,i] ← 1
12:        else
13:            M[0,i] ← 0
14:        end if
15:    end for
16:    for i ← 1 to n-3 do
17:        for j ← 1 to n-2-i do
18:            if s[n-1-i] complements s[j] then
19:                M[i,j] ← M[i-1,j-1]+1
20:            else
21:                M[i,j] ← 0
22:            end if
23:        end for
24:    end for
25:    return M
26: end procedure
```

Por fim, na última etapa é realizada uma extensão para o interior das diagonais esquerda e direita, em cada uma das hastes longas não exatas (Figura 3.7 F) selecionadas na etapa anterior, o que permite encontrar bojos e *loops* não simétricos dos hairpins. É realizada uma busca local na tentativa de encontrar pre-miRNAs de ta-

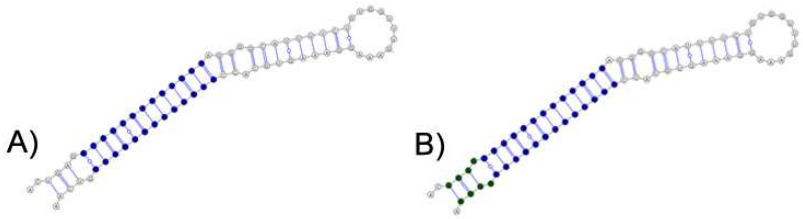

Figura 3.6. Estrutura secundária de um pre-miRNA. (A) Em azul, a maior haste exata. (B) Em verde: extensão da haste exata que resulta em uma haste longa não exata.

manhos mínimo e máximo informados previamente. Cada sequência encontrada tem seus atributos extraídos e as instâncias classificadas (Figura 3.7 G). As sequências com igual ou superior ao ponto de corte informado são selecionadas. (figure 3.7 H).

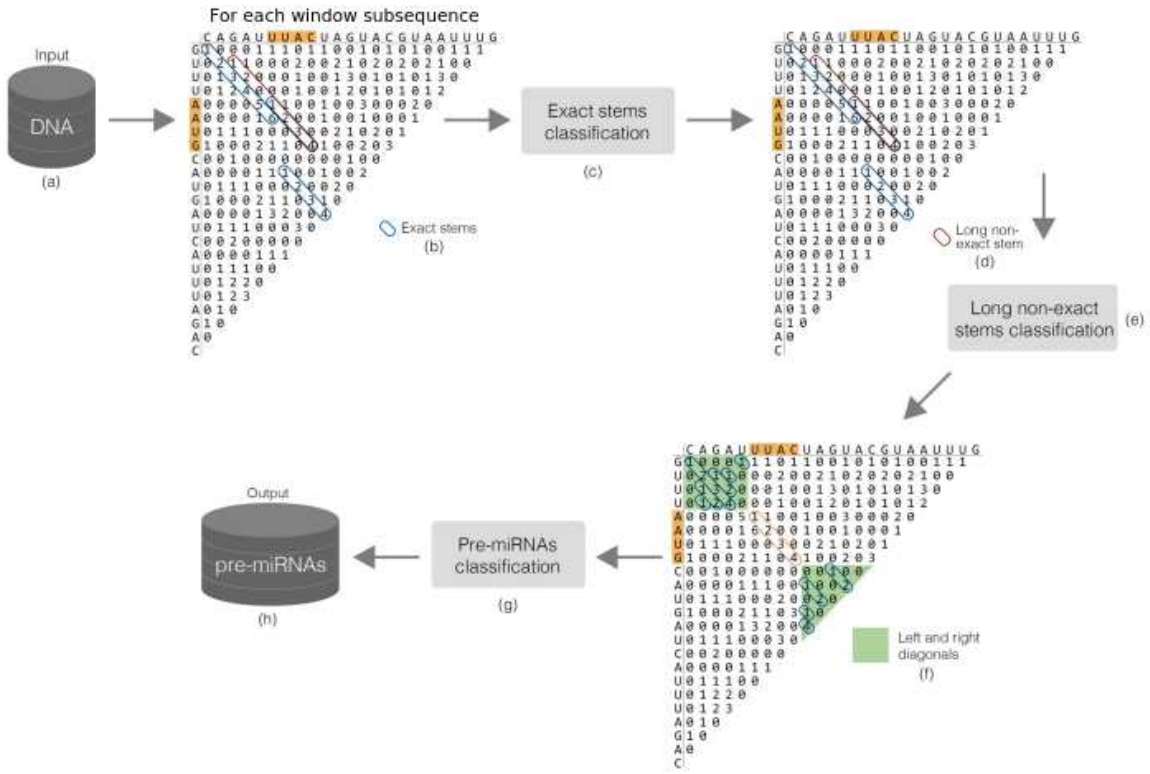

Figura 3.7. Predição de pre-miRNAs. Para uma sequência de DNA informada, é realizada uma busca por pre-miRNAs na matriz de pares de base utilizando três classificações de Aprendizagem de Máquina.

Então, uma nova janela é gerada a partir do deslocamento de 10 nt da janela anterior. Uma nova matriz de pares de bases é gerada e as etapas da busca são executadas novamente. Este processo continua até o fim da sequência de DNA informada. Por

fim, os pre-miRNAs encontrados são mostrados assim como suas posições na sequência de DNA ou RNA informada.

Capítulo 4

Resultados e Discussão

Complexidade do Algoritmo

Como a etapa de treinamento é realizada uma única vez e os modelos resultantes aplicados tanto quanto se necessite, a análise de espaço e tempo não levou em consideração a fase de treinamento.

Acerca do espaço, dada uma sequência de entrada de tamanho n e uma janela deslizante de tamanho m, o espaço requerido pelo algoritmo em função destas variáveis tem complexidade $\theta(n + m^2)$. Nota-se que é necessário armazenar todo o genoma e a matriz triangular de pares de base. Para grandes sequências de entrada, tais como um cromossomo inteiro, ou mesmo um genoma inteiro, é evidente que $n \gg m$. Sendo assim, a complexidade do espaço pode ser considerada como $\theta(n)$.

Em relação ao tempo, para as mesmas variáveis n e m acima, podemos assumir n subsequências de janela para analisar, ou seja, n execuções do procedimento de três estágios, e uma matriz $m \times m$ para explorar em cada caso. Considerando um cenário extremo em cada fase, onde cada uma das m^2 entradas da matriz tem de ser processada, a exploração da matriz tem complexidade $O(m^3)$, pois, para cada posição na matriz, são necessários m acessos adicionais para processar as respectivas diagonais. Portanto, a complexidade de tempo do algoritmo como um todo é $O(nm^3)$.

Os dados

Para comparar nosso método com as ferramentas computacionais atuais, foram utilizados três *test sets* descritos por Tempel & Tahi [2012] :

- Uma sequência artificial do genoma humano com 30500 nt, criada utilizando 100

sequências de pre-miRNAs humano, obtidas do mirBase - release 17 e concatenadas com sequências de genes de mRNAs humano;

- Uma sequência do genoma humano, obtida de um cluster de pre-miRNAs no cromossomo humano 19 (fita +) contendo 50 pre-miRNAs, onde o primeiro começa na posição 54,169,933 e o último termina na posição 54,485,651;
- Uma sequência do genoma do rato obtida de um cluster de pre-miRNAs no cromossomo 2 do rato (fita +) contendo 71 pre-miRNAs, onde o primeiro começa na posição 10,388,290 e o último termina na posição 10,439,906.

Comparando o Mirnacle com outros métodos de predição ab initio de pre-miRNA

Depois de selecionar a mais adequada abordagem de AM, pudemos concluir nossa ferramenta computacional e compará-la com outros métodos propostos anteriormente para a previsão ab initio de pre-miRNAs. Nos experimentos realizados, apenas métodos ab initio da terceira categoria, mencionados anteriormente, foram considerados.

Os parâmetros do Mirnacle são: tamanho mínimo da haste exata, tamanho da janela deslizante, os *threholds* de cada uma das três etapas, o tamanho mínimo do pre-miRNA e tamanho máximo do pré-miRNA. Em todos os experimentos, o tamanho mínimo da haste exata, o tamanho de janela deslizante, o tamanho mínimo pré-miRNA, e o tamanho máximo pre-miRNA foram ajustados para 4, 150, 50, e 150, respectivamente, sendo que o incremento da varredura da sequência foi de 10 nt.

Utilizando o mesmo critério dos autores do miRNAFold, um *hairpin* predito é considerado verdadeiro se a distância do centro do *hairpin* predito e o centro do *hairpin* conhecido for menor ou igual a 10% do tamanho do *hairpin* conhecido. Adicionalmente, para comparar nossos resultados com os resultados da literatura, as métricas estatísticas de sensibilidade e seletividade foram utilizadas. Sensibilidade indica a capacidade do algoritmo de detectar pre-miRNAs reais e seletividade indica a probabilidade de um *hairpin* predito ser real. Podemos obter a seletividade e sensibilidade utilizando as seguintes equações:

$$seletividade = 100 \times \frac{TP}{(TP + FP)},$$

$$sensibilidade = 100 \times \frac{TP}{(TP + FN)},$$

Tabela 4.1. Comparação dos métodos de predição ab initio na sequência artificial humana. Os resultados da seletividade e sensibilidade de MirnaSearch foram tirados do seu artigo e os resultados da seletividade e sensibilidade de miRNAFold, CID-miRNA, miRPara e Vmir foram tirados do artigo miRNAFold.

Método	Sensibilidade	Seletividade	GM	Tempo(mm:ss)
Mirnacle (0.3,0.3,0.7)	97	81.51	88.91	14:58
Mirnacle (0.4,0.4,0.7)	86	85.15	85.57	07:31
Mirnacle (0.5,0.5,0.7)	65	86.76	75.09	03:24
Mirnacle (0.6,0.6,0.7)	55	94.83	72.21	02:05
Mirnacle (0.8,0.8,0.7)	23	95.83	46.94	01:20
MirnaSearch	97	39.34	61.77	*
miRNAFold	97	19.17	43.12	00:84
miRPara	97	9.70	30.67	05:24
CID-miRNA	97	11.72	33.71	90:49
VMir	28	1.32	6.07	02:32

onde *true positive* (TP) é a quantidade de pre-miRNAs que foram preditos corretamente, *false negative* (FN) é a quantidade de pre-miRNAs conhecidos que não foram preditos e *false positive* (FP) é a quantidade de pre-miRNAs preditos que não correspondem a pre-miRNAs reais. Nas Tabelas 4.1, 4.2 e 4.3 , apresentamos os resultados dos programas nas se quências humana artificial, humana e do rato reais, respectivamente.

Da mesma forma que realizados pelo miRNAFold, pontos de corte apropriados para as três fases foram estabelecidos utilizando a sequência humana artificial. Devido a *Random Forest* produzir a probabilidade de um exemplo ser positivo, em vez de uma saída binária, esta probabilidade discriminante pode ser usada de acordo com um objetivo particular. Assim, se a sensibilidade é o mais importante, um ponto de corte baixo deve ser usado. Por outro lado, se a seletividade é prioridade, por exemplo, para minimizar as validações de laboratório, um ponto de corte elevado é mais apropriado. Sendo assim, é necessário definir os ponto de corte de cada modelo utilizado para cada fase. Para este fim, tentamos várias combinações de ponto de corte (não mostradas) e cinco delas foram selecionadas (representadas por triplas ordenadas).

Pode ser visto na Tabela 4.1 que ponto de corte altos resultaram em alta seletividade e diminuição da Sensibilidade. Por outro lado, pontos de corte baixos retornaram alta sensibilidade e diminuição da seletividade. Com ponto de corte de 0,3 na primeira etapa, 0,3 na segunda e 0,7 na terceira, foi possível encontrar um equilíbrio entre a seletividade e sensibilidade (88.91 % de média geométrica). Por consequência disto, estes pontos de corte foram utilizados nos demais conjuntos de teste.

No que diz respeito ao tempo de execução, o desempenho do Mirnacle foi al-

Tabela 4.2. Comparação das predições na sequência humana real. Os resultados da seletividade e sensibilidade do MirnaSearch foram tirados do seu artigo e os resultados da seletividade e sensibilidade de miRNAFold, CID-miRNA, miRPara e Vmir foram tirados do artigo do miRNAFold.

Método	Sensibilidade	Seletividade	M.G.
Mirnacle	100	29.52	54.33
MirnaSearch	100	1.43	11.95
miRNAFold	100	0.89	9.43
miRPara	98	0.93	9.54
CID-miRNA	38	0.69	5.12
VMir	100	0.56	7.48

tamente variável. Isto é porque pontos de corte baixos na primeira e segunda fases significam um filtro menos rígido das hastes exatas e não exatas, isto é, a terceira fase irá conter mais sequências a serem estendidas a fim de completar o *hairpin*, o que acarreta um maior tempo de execução. Pontos de corte altos na primeira e segunda fases, por outro lado, significam menos hastes não exatas para expandir, acelerando o processo. Além disto, a exploração da matriz na terceira fase para inspecionar diferentes possibilidades de um *hairpin* completo é a parte computacionalmente mais cara. Comparando a execução Mirnacle que levou 14 minutos e 58 segundos, com o tempo de execução de outros métodos, Mirnacle só supera CID-miRNA. No entanto, considerando que pudemos melhorar substancialmente a seletividade, o tempo economizado em experimentos de laboratório é provavelmente mais significativo. Como pode ser visto, o tempo gasto pelo *webserver* MirnaSearch não foi relatado, pois seus autores não mencionam qualquer experimento para medir o tempo. Além disso, MirnaSearch está disponível apenas como um servidor web, o que torna inviável medir o seu tempo de execução de uma forma justa.

Adicionalmente, na sequência humana real, a Tabela 4.2 mostra que a sensibilidade e seletividade do Mirnacle supera MirnaSearch, miRNAFold, CID-miRNA, miRPara e VMir. O Mirnacle foi 20 vezes mais seletivo do que o *webserver* MirnaSearch.

Finalmente, na sequência genômica real do *Mus musculus*, a Tabela 4.3 mostra que a sensibilidade e a seletividade do Mirnacle supera o MirnaSearch, miRNAFold, CID-miRNA, miRPara e VMir. Não foi possível recuperar apenas um pre-miRNA conhecido (98.61 % de sensibilidade) na sequência. Embora o MirnaSearch tenha conseguido encontrar todos os pre-miRNAs, nossa abordagem apresentou uma seletividade 11 vezes superior. Por consequência disto, este aumento de seletividade produzido pelo

Tabela 4.3. Resultados da predição na sequência do rato real. Os resultados da seletividade e sensibilidade do miRNAFold, CID-miRNA, miRPara e Vmir foram tirados do artigo do miRNAFold.

	Sensibilidade	Seletividade	GM
Mirnacle	98.61	47.33	68.31
MirnaSearch	100	4.13	20.32
miRNAFold	98.59	7.71	27.57
miRPara	98.59	5.34	22.94
CID-miRNA	29.58	0.82	4.92
VMir	88.73	2.93	16.12

Mirnacle conduziu uma média geométrica muito superior. Adicionalmente, nossa abordagem obteve 6 vezes mais seletividade do que o método miRNAfold.

Embora tenha apresentado resultados superiores aos da literatura, por outro lado, o método de janela deslizante para predição de pre-miRNAs na sequência genômica real do rato tem um problema com sequências de pre-miRNAs que aparecem mais de uma vez ao longo da sequência completa. Por consequência disto, as sequências dos pre-miRNAs MI0005502, MI0002402 e MI0004523 aparecem seis, oito e onze vezes respectivamente. Como resultado, pseudo pre-miRNAs são falsamente preditos nesses casos.

Adicionalmente, os pontos de corte manuais possibilitam ao usuário privilegiar seletividade ou sensibilidade na predição de novos pre-miRNAs.

Por fim, diferentemente das demais abordagens, nosso software permite que o usuário informe novos exemplos positivos e negativos, a fim de gerar modelos e procurar por novos pre-miRNAs em qualquer eucarioto utilizando seus próprios modelos personalizados.

Capítulo 5

Conclusões

Neste trabalho, apresentamos um novo método para predição ab initio de pre-miRNAs através de aprendizagem de máquina. O Mirnacle explora a utilização de aprendizagem de máquina para maximizar a seletividade da estratégia desenvolvida por Tempel & Tahi [2012].

A partir de uma sequência de DNA, o primeiro passo é localizar hastes exatas em uma janela de pares de bases, classificá-las e selecionar as hastes exatas mais prováveis. Então, é feita uma extensão em cada haste exata para localizar, classificar e selecionar as hastes longas não exatas mais prováveis. Por fim, é realizada uma extensão em cada haste longa não exata para localizar e classificar *hairp*ins. Aqueles possíveis pre-miRNAs que atendam o ponto de corte final são mostrados, assim como suas posições na sequência estudada.

O método foi testado nas sequências de *Homo sapiens* e *Mus musculus* e comparado com os métodos encontrados na literatura. Como resultado, nosso método obteve uma melhor seletividade do que todos os outros métodos, com sensibilidade próxima ou melhor.

É importante notar que a seletividade pôde ser melhorada sem que ocorresse a perda de sensibilidade. Novas técnicas de aprendizagem de máquina e novos atributos podem ser testados visando uma melhora de seletividade e sensibilidade.

Em trabalhos futuros, sugere-se a obtenção automática dos melhores pontos de corte para cada fase do algoritmo. Pode-se também realizar a experimentação em outras espécies de eucariotos.

Além disto, visando obter ganhos de seletividade, poderão ser adicionadas buscas por miRNAs contidos nos pre-miRNAs preditos por este trabalho.

Por fim, implementar os cálculos de energia livre mínima e delta G dentro do próprio Mirnacle ao invés de realizar chamadas externas ao pacote Vienna, além de pa-

ralelizar o processamento com GPU (*graphics processor unit*), certamente irá diminuir o tempo de execução do método proposto neste trabalho.

Referências Bibliográficas

Alberts, B.; Johnson, A.; Lewis, J.; Raff, M.; Roberts, K. & Walter, P. (2010). *Biologia molecular da célula*. Artmed.

Ambros, V. (2004). The functions of animal microrna. *Nature*, 431(7006):350--355.

Bartel, D. P. (2004). Microrna: genomics, biogenesis, mechanism, and function. *cell*, 116(2):281--297.

Batuwita, R. & Palade, V. (2009). micropred: effective classification of pre-mirnas for human mirna gene prediction. *Bioinformatics*, 25(8):989--995.

Brennecke, J.; Hipfner, D. R.; Stark, A.; Russell, R. B. & Cohen, S. M. (2003). bantam encodes a developmentally regulated microrna that controls cell proliferation and regulates the proapoptotic gene hid in drosophila. *Cell*, 113(1):25--36.

Carrington, J. C. & Ambros, V. (2003). Role of microrna in plant and animal development. *Science*, 301(5631):336--338.

Cerqueira, F. R.; Ferreira, T. G.; de Paiva Oliveira, A.; Augusto, D. A.; Krempser, E.; Barbosa, H. J. C.; Franceschini, S. d. C. C.; de Freitas, B. A. C.; Gomes, A. P. & Siqueira-Batista, R. (2014). Nicesim: an open-source simulator based on machine learning techniques to support medical research on prenatal and perinatal care decision making. *Artificial intelligence in medicine*, 62(3):193--201.

Chawla, N. V.; Bowyer, K. W.; Hall, L. O. & Kegelmeyer, W. P. (2011). Smote: synthetic minority over-sampling technique. *arXiv preprint arXiv:1106.1813*.

Chen, J.-F.; Mandel, E. M.; Thomson, J. M.; Wu, Q.; Callis, T. E.; Hammond, S. M.; Conlon, F. L. & Wang, D.-Z. (2005). The role of microrna-1 and microrna-133 in skeletal muscle proliferation and differentiation. *Nature genetics*, 38(2):228--233.

Chiu, D. K. & Kolodziejczak, T. (1991). Inferring consensus structure from nucleic acid sequences. *Computer applications in the biosciences: CABIOS*, 7(3):347--352.

Collins, F. S.; Green, E. D.; Guttmacher, A. E. & Guyer, M. S. (2003). A vision for the future of genomics research. *Nature*, 422(6934):835--847.

Dougherty, G. (2012). *Pattern recognition and classification: an introduction*. Springer Science & Business Media.

Fridell, R. (2005). *Decoding life: unraveling the mysteries of the genome*. Twenty-First Century Books.

Genetics, H. R. (2013). Gene expression.

Gregory, T. R. (2011). *The evolution of the genome*. Academic Press.

Griffiths-Jones, S.; Grocock, R. J.; Van Dongen, S.; Bateman, A. & Enright, A. J. (2006). mirbase: microrna sequences, targets and gene nomenclature. *Nucleic acids research*, 34(suppl 1):D140--D144.

Grundhoff, A.; Sullivan, C. S. & Ganem, D. (2006). A combined computational and microarray-based approach identifies novel micrornas encoded by human gamma-herpesviruses. *Rna*, 12(5):733--750.

Gudyś, A.; Szcześniak, M. W.; Sikora, M. & Makałowska, I. (2013). Huntmi: an efficient and taxon-specific approach in pre-mirna identification. *BMC bioinformatics*, 14(1):83.

Ha, M. & Kim, V. N. (2014). Regulation of microrna biogenesis. *Nature Reviews Molecular Cell Biology*, 15(8):509--524.

Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P. & Witten, I. H. (2009). The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10--18.

Hertel, J. & Stadler, P. F. (2006). Hairpins in a haystack: recognizing microrna precursors in comparative genomics data. *Bioinformatics*, 22(14):e197--e202.

Hofacker, I. L. (2003). Vienna rna secondary structure server. *Nucleic acids research*, 31(13):3429--3431.

Huang, T.-H.; Fan, B.; Rothschild, M. F.; Hu, Z.-L.; Li, K. & Zhao, S.-H. (2007). Mirfinder: an improved approach and software implementation for genome-wide fast microrna precursor scans. *BMC bioinformatics*, 8(1):341.

Inza, I.; Calvo, B.; Armañanzas, R.; Bengoetxea, E.; Larrañaga, P. & Lozano, J. A. (2010). Machine learning: an indispensable tool in bioinformatics. Em *Bioinformatics methods in clinical research*, pp. 25--48. Springer.

Jiang, P.; Wu, H.; Wang, W.; Ma, W.; Sun, X. & Lu, Z. (2007). Mipred: classification of real and pseudo microrna precursors using random forest prediction model with combined features. *Nucleic acids research*, 35(suppl 2):W339--W344.

Krol, J.; Loedige, I. & Filipowicz, W. (2010). The widespread regulation of microrna biogenesis, function and decay. *Nature Reviews Genetics*, 11(9):597--610.

Lai, E. C.; Tomancak, P.; Williams, R. W.; Rubin, G. M. et al. (2003). Computational identification of drosophila microrna genes. *Genome Biol*, 4(7):R42.

Lambert, A.; Fontaine, J.-F.; Legendre, M.; Leclerc, F.; Permal, E.; Major, F.; Putzer, H.; Delfour, O.; Michot, B. & Gautheret, D. (2004). The erpin server: an interface to profile-based rna motif identification. *Nucleic acids research*, 32(suppl 2):W160--W165.

Lawrie, C. H. (2013). Micrornas: A brief introduction. *MicroRNAs in Medicine*, pp. 1--24.

Lertampaiporn, S.; Thammarongtham, C.; Nukoolkit, C.; Kaewkamnerdpong, B. & Ruengjitchatchawalya, M. (2013). Heterogeneous ensemble approach with discriminative features and modified-smotebagging for pre-mirna classification. *Nucleic acids research*, 41(1):e21--e21.

Lim, L. P.; Glasner, M. E.; Yekta, S.; Burge, C. B. & Bartel, D. P. (2003). Vertebrate microrna genes. *Science*, 299(5612):1540--1540.

Liu, B.; Fang, L.; Chen, J.; Liu, F. & Wang, X. (2015). mirna-dis: microrna precursor identification based on distance structure status pairs. *Molecular BioSystems*, 11(4):1194--1204.

Lockhart, D. J. & Winzeler, E. A. (2000). Genomics, gene expression and dna arrays. *nature*, 405(6788):827--836.

Markham, N. R. & Zuker, M. (2008). Unafold. Em *Bioinformatics*, pp. 3--31. Springer.

Murchison, E. P.; Stein, P.; Xuan, Z.; Pan, H.; Zhang, M. Q.; Schultz, R. M. & Hannon, G. J. (2007). Critical roles for dicer in the female germline. *Genes & development*, 21(6):682--693.

Nature, E. (2010). Gene expression.

Nelson, D. L.; Lehninger, A. L. & Cox, M. M. (2008). *Lehninger principles of biochemistry*. Macmillan.

Osada, H. & Takahashi, T. (2007). Micrornas in biological processes and carcinogenesis. *Carcinogenesis*, 28(1):2--12.

Pang-Ning, T.; Steinbach, M.; Kumar, V. et al. (2005). *Introduction to data mining*. Pearson.

Paschoal, A. R.; Maracaja-Coutinho, V.; Setubal, J. C.; Simões, Z. L. P.; Verjovski-Almeida, S. & Durham, A. M. (2012). Non-coding transcription characterization and annotation. *RNA biology*, 9(3).

Pevsner, J. (2009). *Bioinformatics and functional genomics*. John Wiley & Sons.

Poy, M. N.; Eliasson, L.; Krutzfeldt, J.; Kuwajima, S.; Ma, X.; MacDonald, P. E.; Pfeffer, S.; Tuschl, T.; Rajewsky, N.; Rorsman, P. et al. (2004). A pancreatic islet-specific microrna regulates insulin secretion. *Nature*, 432(7014):226--230.

Russell, P. J. (2010). *IGenetics*. Benjamin Cummings San Francisco.

Shivdasani, R. A. (2006). Micrornas: regulators of gene expression and cell differentiation. *Blood*, 108(12):3646--3653.

Suh, M.-R.; Lee, Y.; Kim, J. Y.; Kim, S.-K.; Moon, S.-H.; Lee, J. Y.; Cha, K.-Y.; Chung, H. M.; Yoon, H. S.; Moon, S. Y. et al. (2004). Human embryonic stem cells express a unique set of micrornas. *Developmental biology*, 270(2):488--498.

Tan, A. C.; Gilbert, D. et al. (2001). Machine learning and its application to bioinformatics: an overview. *Bioinformatics*.

Tempel, S. & Tahi, F. (2012). A fast ab-initio method for predicting mirna precursors in genomes. *Nucleic acids research*, 40(11):e80--e80.

Tempel, S.; Zerath, B.; Zehraoui, F.; Tahi, F. et al. (2015). mirboost: boosting support vector machines for microrna precursor classification. *RNA*.

Terai, G.; Komori, T.; Asai, K. & Kin, T. (2007). mirrim: a novel system to find conserved mirnas with high sensitivity and specificity. *Rna*, 13(12):2081--2090.

Thangadurai, D. & Sangeetha, J. (2014). *Biotechnology and Bioinformatics: Advances and Applications for Bioenergy, Bioremediation and Biopharmaceutical Research*. CRC Press.

Titov, I. I.; Vorobiev, D. G.; Ivanisenko, V. A. & Kolchanov, N. A. (2002). A fast genetic algorithm for rna secondary structure analysis. *Russian chemical bulletin*, 51(7):1135--1144.

Titov, I. I. & Vorozheykin, P. S. (2013). Ab initio human mirna and pre-mirna prediction. *Journal of bioinformatics and computational biology*, 11(06).

Tyagi, S.; Vaz, C.; Gupta, V.; Bhatia, R.; Maheshwari, S.; Srinivasan, A. & Bhattacharya, A. (2008). Cid-mirna: a web server for prediction of novel mirna precursors in human genome. *Biochemical and biophysical research communications*, 372(4):831--834.

Wang, X.; Zhang, J.; Li, F.; Gu, J.; He, T.; Zhang, X. & Li, Y. (2005). Microrna identification based on sequence and structure alignment. *Bioinformatics*, 21(18):3610--3614.

Wilfred, B. R.; Wang, W.-X. & Nelson, P. T. (2007). Energizing mirna research: a review of the role of mirnas in lipid metabolism, with a prediction that mir-103/107 regulates human metabolic pathways. *Molecular genetics and metabolism*, 91(3):209--217.

Williams, A. H.; Liu, N.; van Rooij, E. & Olson, E. N. (2009). Microrna control of muscle development and disease. *Current opinion in cell biology*, 21(3):461--469.

Witten, I. H. & Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

Wu, Y.; Wei, B.; Liu, H.; Li, T. & Rayner, S. (2011). Mirpara: a svm-based software tool for prediction of most probable microrna coding regions in genome scale sequences. *BMC bioinformatics*, 12(1):107.

Xia, X. (2007). *Bioinformatics and the cell: modern computational approaches in genomics, proteomics and transcriptomics*. Springer Science & Business Media.

Xuan, P.; Guo, M.; Liu, X.; Huang, Y.; Li, W. & Huang, Y. (2011). Plantmirnapred: efficient classification of real and pseudo plant pre-mirnas. *Bioinformatics*, 27(10):1368--1376.

Xue, C.; Li, F.; He, T.; Liu, G.-P.; Li, Y. & Zhang, X. (2005). Classification of real and pseudo microrna precursors using local structure-sequence features and support vector machine. *BMC bioinformatics*, 6(1):310.

Yousef, M.; Nebozhyn, M.; Shatkay, H.; Kanterakis, S.; Showe, L. C. & Showe, M. K. (2006). Combining multi-species genomic data for microrna identification using a naive bayes classifier. *Bioinformatics*, 22(11):1325--1334.

Zare, H.; Khodursky, A. & Sartorelli, V. (2014). An evolutionarily biased distribution of mirna sites toward regulatory genes with high promoter-driven intrinsic transcriptional noise. *BMC evolutionary biology*, 14(1):74.

Zhang, Y. & Rajapakse, J. C. (2009). *Machine learning in bioinformatics*, volume 4. John Wiley & Sons.

Zuker, M. (2003). Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic acids research*, 31(13):3406--3415.

Apêndice A

Poster publicado no evento ISCB-Latin America x-Meeting on Bioinformatics with BSB and SoiBio

Os resultados iniciais deste trabalho foram publicados no formato de poster em 2014 no evento ISCB-Latin America x-Meeting on Bioinformatics with BSB and SoiBio, em Belo Horizonte/MG. Intitulado "Machine Learning Techniques to Increase Selectivity in pre-miRNA Prediction", além de ncRNAs, o trabalho utilizava subsequências mRNAs como exemplos negativos nos conjuntos de treinamento.

MACHINE LEARNING TECHNIQUES TO INCREASE SELECTIVITY IN PRE-MIRNA PREDICTION

Cerqueira F R[1], Marques Y B[1], Oliveira A P[1]

[1]Department of Informatics, Universidade Federal de Viçosa, Brazil.

ABSTRACT

MiRNAs are small non-coding RNAs (approximately 22 nt) that regulate gene expression by binding to a specific mRNA target and promoting its degradation and / or translation inhibition. A strategy to find new miRNAs is to search for its precursors (pre-miRNAs), which are slightly larger structures (70-120 nt) and have a hairpin structural form. However, characterizing pre-miRNAs in vivo is still a complex task. As a consequence, in silico methods were developed to predict the genomic location of pre-miRNAs. Nevertheless, the current computational tools have problems of selectivity. This work presents an extension of the method developed by Tempel et al., 2012, with the aim of improving selectivity through machine learning techniques combined with the SMOTE method that copes with imbalance datasets. We used two datasets built from the human genome, and one dataset built from *Mus musculus* to compare our method with other important approaches in the literature. The results have shown that our procedures could substantially improve selectivity without compromising sensibility.

INTRODUCTION

MicroRNAs (miRNAs) are recognized as key regulators of gene expression in plants and animals. Thus, miRNAs are involved in the majority of biological process, making the study of these molecules one of the most relevant topics of molecular biology nowadays.

This work presents an extension of the method developed by Tempel et al., 2012, using machine learning (ML) techniques with the aim of improving selectivity. We use 85 key attributes to build ML models through the random forest approach combined with the SMOTE (synthetic minority over-sampling technique) method that copes with imbalance datasets. The ML models were obtained from datasets where positive examples were taken from miRBase and negative examples were composed of non-coding RNAs (except for miRNAs) as well as mRNAs of the species for which the experiment was (Figure 2).

In order to predict new pre-miRNAs for a given sequence of DNA (Figure 3a), the first step is to search for exact stems in the base-pair matrix (Figure 3b), classify, and select the most likely (Figure 3c). Next, extend the search for long non exact stems (Figure 3d), classify, and select the most likely (Figure 3e). Finally, search for pre-miRNAs among each previous selected non-exact stem (Figure 3f). Then, pseudo-hairpins are classified (Figure 3g) and those that meet the threshold are returned as well as their positions in the studied sequence (Figure 3h).

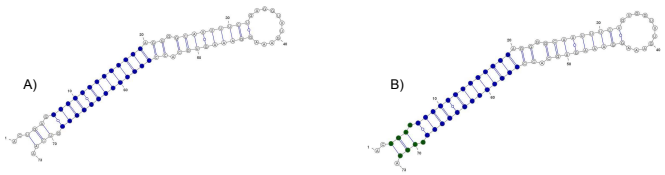

Figure 1: pre-miRNA secondary structure. (A) In blue, the biggest exact stem. (B) Extension for long non-exact stem is shown in green.

Pseudocode for searching for pre-miRNAs

```
For each DNA window
    Generate base pairing matrix
    Find the Exact Stems
    Classify the Exact Stems
    Select the Exact Stems that meet Threshold 1
    For each Exact Stem selected in the previous step
      Extend the search for Long Non Exact Stems
      Classify the Long Non Exact Stems
      Select the Long Non Exact Stems that meet Threshold 2
      For each Long Non Exact Stem selected in the previous step
        Perform local search in order to find a pre-miRNA
        Return the position of a pre-miRNA that meets Threshold 3
```

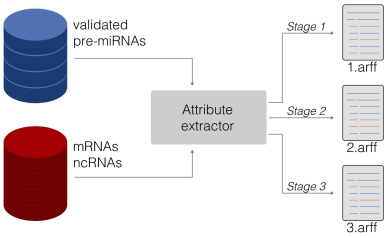

Figure 2: Example of machine learning training. Real pre-miRNAs are used as positive examples. For negative examples, mRNAs and ncRNAs (except miRNAs) are used.

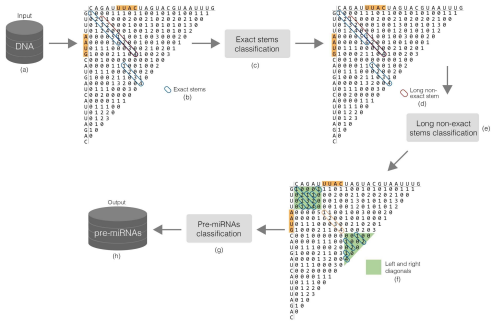

Figure 3: pre-miRNA Prediction. For a given sequence of DNA, a search for a pre-miRNA in the base-pairing matrix and the three ML classifications.

RESULTS

To compare our method with current computational tools, we measured sensitivity and selectivity for three datasets used in the work of Tempel and colleagues.

Table 1: Human: Artificial

Method	Sen.	Sel.
Our Method	**100**	**75.37**
MirnaSearch	97	39.34
miRNAFold	97	19.17
miRPara	97	9.70
CID-miRNA	97	11.72
Vmir	28	1.32

Table 2: Human: Real

Method	Sen.	Sel.
Our Method	**100**	**7.90**
MirnaSearch	100	1.43
miRNAFold	100	0.83
miRPara	90	9.70
SSCProfiler	12	11.72
CID-miRNA	38	1.32
Vmir	100	0.56

Table 3: *Mus musculus*

Method	Sen.	Sel.
Our Method	**100**	**33.90**
MirnaSearch	100	4.13
miRNAFold	98.59	7.71
miRPara	98.59	5.34
CID-miRNA	29.58	0.82
Vmir	88.73	1.32

Comparing our method with other important approaches in the literature, we have shown that our procedures could substantially improve selectivity without compromising sensibility. For three datasets used in our experiments, our method predicted all pre-miRNAs (100% sensitivity) and could deliver an increase in selectivity of two-fold, 8-fold, and 5-fold, respectively, compared with the best computational tools.

CONCLUSION

This work presents an extension of the method developed by Tempel et al., 2012, with the aim of improving selectivity through machine learning techniques combined with the SMOTE method that copes with imbalance datasets. We used two datasets built from the human genome, and one dataset built from *Mus musculus* to compare our method with other important approaches in the literature. The results have shown that our procedures could substantially improve selectivity without compromising sensibility

ACKNOWLEDGEMENTS

This work is supported by CAPES, CNPq, FAPEMIG, and IFNMG

Apêndice B

Full Paper publicado no evento X-Meeting 2015 - 11th International Conference of the AB3C + Brazilian Symposium of Bioinformatics

O artigo completo intitulado "Mirnacle: Machine learning with SMOTE and random forest for improving selectivity in pre-miRNA ab initio prediction"foi apresentado como *Full Paper* no evento X-Meeting 2015 - 11th International Conference of the AB3C + Brazilian Symposium of Bioinformatics em São Paulo/SP e aceito na revista BMC Bioinformatics(versão final em desenvolvimento).

RESEARCH

Mirnacle: Machine learning with SMOTE and random forest for improving selectivity in pre-miRNA ab initio prediction

Yuri B Marques[1,2], Alcione P Oliveira[1,3], Ana Tereza R Vasconcelos[4] and Fabio R Cerqueira[1*]

*Correspondence:
fabio.cerqueira@ufv.br
[1] Department of Informatics, Universidade Federal de Viçosa, 36570-900 Viçosa, Brazil
Full list of author information is available at the end of the article

Abstract

Background:
MicroRNAs (miRNAs) are key gene expression regulators in plants and animals. Therefore, miRNAs are involved in several biological processes, making the study of these molecules one of the most relevant topics of molecular biology nowadays. However, characterizing miRNAs in vivo is still a complex task. As a consequence, in silico methods have been developed to predict miRNA loci. A common ab initio strategy to find miRNAs in genomic data is to search for sequences that can fold into the typical hairpin structure of miRNA precursors (pre-miRNAs). The current ab initio approaches, however, have selectivity issues, i.e., a high number of false positives is reported, which can lead to laborious and costly attempts to provide biological validation. This study presents an extension of the ab initio method miRNAFold, with the aim of improving selectivity through machine learning techniques, namely, random forest combined with the SMOTE procedure that copes with imbalance datasets.

Results:
By comparing our method, termed Mirnacle, with other important approaches in the literature, we demonstrate that Mirnacle substantially improves selectivity without compromising sensitivity. For the three datasets used in our experiments, our method achieved at least 97% of sensitivity and could deliver a two-fold, 20-fold, and 6-fold increase in selectivity, respectively, compared with the best results of current computational tools.

Conclusions:
The extension of miRNAFold by the introduction of machine learning techniques, significantly increases selectivity in pre-miRNA ab initio prediction, which optimally contributes to advanced studies on miRNAs, as the need of biological validations is diminished. Hopefully, new research, such as studies of severe diseases caused by miRNA malfunction, will benefit from this powerful computational tool.

Keywords: pre-miRNA ab initio prediction; random forest; smote; microRNA; machine learning; data mining

Background

MicroRNAs (miRNAs) constitute currently a major research topic in molecular biology [1]. These molecules are responsible for post-transcriptional regulation of gene expression, and are involved in several biological processes such as physiological roles in animals and plants [2], embryonic differentiation [3], skeletal muscle

development [4], several cardiovascular disorders [5], expansion of skin stem cells [6], hematopoiesis [7], control of proliferation and death of cells [8], insulin secretion [9], adipogenesis and obesity [10], and diseases such as cancer [11].

In the miRNA biogenesis, the corresponding gene is transcribed into a primary miRNA (pri-miRNA) that undergoes cleavage by the Drosha complex, resulting in a hairpin-shaped miRNA precursor (pre-miRNA) of approximately 70 nt. Next, the precursor is exported to the cytoplasm by the protein Exportin5 and cleaved into the mature miRNA by the enzyme Dicer [1].

Identifying miRNA loci in vivo is an expensive and complex task. As a result, computational tools have been developed to perform in silico predictions. Comparative approaches, i.e., methods that use conservation of functional genomic elements of phylogenetically-close species, are a natural means for carrying out predictions, because many miRNAs are well conserved among eukaryotes [12]. Additionally, methods based on next-generation sequencing (NGS) data are of increasing interest, as they deliver good sensitivity and provide the possibility of analyzing expression profiles [13]. Another important computational solution that complements the approaches mentioned above, and which is a good alternative for identifying species-specific miRNAs, is the so-called ab initio approach.

According to Tempel and Tahi [14], ab initio methods can be classified into three categories: 1) Methods whose input is a putative pre-miRNA sequence and that classify the given candidate as true or false; 2) methods whose input is comprised of a genomic sequence as well as some other information that helps to predict pre-miRNAs in the given sequence; 3) methods whose input is a genomic sequence and that use no external information to predict pre-miRNAs in the given sequence. Tempel and Tahi call the third category completely ab inito methods.

Ab initio methods are very useful when no closely-related species with identified miRNAs are available. Even when such an information is accessible, ab initio approaches are important for predicting pre-miRNAs that are specific to the studied genome. Furthermore, NGS still involves a considerable cost and laboratory infrastructure that are not the reality of a significant part of research groups worldwide. Our work matches the third category of ab initio methods. Some important approaches in this category have been previously proposed.

VMir was conceived to predict pre-miRNAs in viruses [15]. Initially, this method processes the input sequence through a sliding window to produce subsequences with length close to the expected length of a pre-miRNA. Then, in order to build a putative secondary structure for each subsequence, VMir uses the software RNAfold [16]. Finally, VMir calculates a score for each pre-miRNA candidate based on selected secondary structure attributes. Only those pre-miRNA candidates with scores above an informed threshold value are given as output.

CID-miRNA applies a stochastic context-free grammar built from secondary structures of known human pre-miRNAs, along with a J48 decision tree to predict pre-miRNAs in a given DNA sequence [17]. To learn the characteristics of negative cases in its model, CID-miRNA uses human ribosomal RNA.

Virgo predicts pre-miRNAs using a variant of the learning algorithm support vector machine (SVM) called $\text{SVM}^{\text{light}}$ [18, 19]. For training the classifier, validated human miRNA precursors are used as positive instances, while negative instances

are artificially produced using coding regions of genes. Virgo extracts several subsequences from the input sequence, and computationally folds them with the software RNAfold. A filter is then applied to select the secondary structures that match known characteristics of pre-miRNAs. Finally, several attributes are extracted from the selected candidates so that the SVM model can assign them a final classification.

The method miRPara also applies SVM for performing classification, and provides the prediction of mature miRNAs in addition to their precursors [20]. Initially, miRPara breaks the input sequence into 500-nt subsequences with a 200-nt overlap. Next, the subsequences are given as input to the software UNAFold for extracting hairpins [21]. Many of the obtained hairpins are then discarded by the application of a filter, and the remaining ones are further analyzed with UNAFold. At last, several attributes related to physical characteristics of miRNAs/pre-miRNAs are extracted from the final set of hairpins for predicting the most likely candidates. To construct the training sets, positive examples were extracted from validated miRNAs/pre-miRNAs, and negative examples were artificially produced from validated pri-miRNAs with the start position of their miRNAs randomly shifted within the sequences.

The method miRNAFold is based on a statistical approach defined from a previous analysis the authors made of pre-miRNAs contained in the miRBase database [14, 22]. Similarly to other methods, miRNAFold analyzes fixed-length sequence windows, searching for putative pre-miRNAs. For each window, miRNAFold constructs the most likely secondary structure without the use of third-party computational tools, in order to speed up the prediction procedure. This is accomplished by the construction of a base-pairing matrix for each window, which is analyzed in three stages. For each stage, several characteristics of parts of putative hairpins are observed according to the statistical study made previously. The final hairpins that match a certain percentage of these characteristics are given as likely pre-miRNA candidates.

The method of Titov and Vorozheykin (TVM) proposes a context-structural Markov model and a hidden Markov model for ab initio prediction of pre-miRNAs and miRNAs, respectively [23]. The authors used 1,872 human pre-miRNAs from the miRBase database as positive instances, and 1,872 random sequences from the pseudo-precursors constructed by Xue and colleagues as negative instances [24]. TVM uses the software GArna to produce secondary structures from the subsequences extracted from the input sequence [25]. Then, several patterns of the primary and secondary structures are evaluated with the proposed models.

Even though the methods described above have provided important contributions, they deliver very low selectivity (also known as precision), i.e., a high number of false positives (FPs) are produced, which means more work at the laboratory bench. In particular, the statistical study made for the method miRNAFold identified secondary structure characteristics of positive instances only, i.e., the method was based on pre-miRNAs alone. The authors did not analyze negative instances in their work. As a consequence, even reaching excellent sensitivity, large amounts of FPs are produced.

In this work, we propose an extension of miRNAFold, which includes negative examples and applies machine learning (ML) techniques to improve selectivity. We

compared our approach with miRNAFold and all other methods mentioned above. Experiments with three datasets demonstrate that we preserved very high sensitivity, while substantially increasing selectivity. Our method, termed Mirnacle, provided an improvement of two-fold, 20-fold, and 6-fold in selectivity, for the respective datasets, compared with the best results of the other approaches.

Methods

Our aim is to predict new pre-miRNAs from a DNA sequence without using any additional information (completely ab initio approach). For each subsequence extracted from the input sequence, Mirnacle executes three stages to try to build the most likely hairpin structure. Each stage classifies parts of the hairpin that is incrementally built. The steps are similar to the miRNAFold approach. One major difference is that we consider negative examples in addition to positive instances (previously identified pre-miRNAs). Another important distinction is the of use ML techniques to perform a more sophisticated classification, where each stage has its own model, with the aim of minimizing the number of false positives.

Ab initio prediction

Similarly to many methods in the ab initio category, Mirnacle applies a sliding window to the informed DNA sequence to obtain subsequences that may represent pre-miRNAs. For each extracted subsequence, the goal is to fold it into a typical pre-miRNA hairpin structure. Third-party programs are often used for this end. However, analogously to miRNAFold, putative foldings are represented here by a triangular base pairing matrix, and the most likely hairpin is obtained from a three-stages analysis of this matrix. Figure 1 illustrates the general structure of our method by showing an example of such a matrix.

To build the matrix for a given sequence $s[0..n-1]$, a column i represents the character $s[i]$, a row i represents the character $s[n-1-i]$, and an entry i, j is a positive integer number if the corresponding bases are complementary, or zero, otherwise. The positive numbers indicate the extension of the paired region. Algorithm 1 clearly describes how the base pairing matrix is constructed. Lines 4-15 initialize the first column and the first row, while lines 16-24 fill the other entries.

The general idea is to build secondary structures incrementally. Minor parts of possible hairpins are identified and then extended to form the complete structures, as can be seen in Figure 2. Initially, long regions of paired bases, termed exact stems, are sought (initial steps shown in Figure 1B/C and illustrated in Figure 2A). Next, the long exact stems found in the previous step are extended to non-exact stems, i.e., parts of a hairpin composed of paired bases interposed between unpaired regions (steps shown in parts D and E of Figure 1 and depicted in Figure 2B). These unpaired regions are symmetrical loops whose size is less than the length of the surrounding exact stems. The extension of an exact stem is achieved by taking into account only its diagonal in the base pairing matrix. Each resultant non-exact stem is considered a good approximation of a hairpin and is thus used at the last stage as the basis for achieving a pre-miRNA secondary structure (the final steps shown in Figure 1F/G). For each non-exact stem, the exact stem that gave rise to it is fixed and other diagonals are explored to make possible the occurrence of asymmetrical internal loops (see Figure 1F).

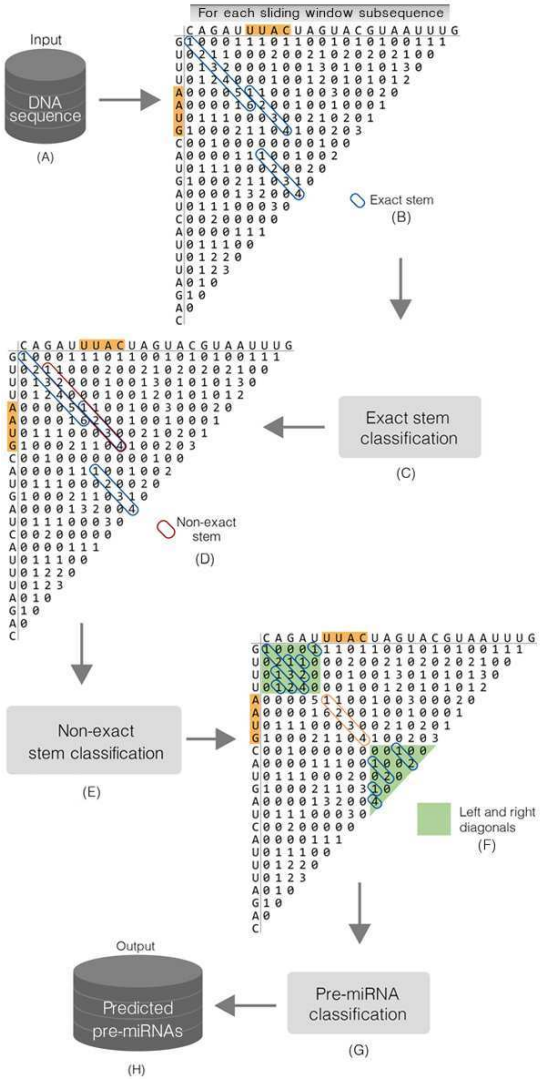

Figure 1 General view of the Mirnacle approach (Adapted from Tempel and Tahi [14]). Given the input DNA sequence (A), a sliding window is used to extract subsequences of length close to the expected pre-miRNA length. For each subsequence, a triangular base pairing matrix (example for the sequence CAGAUUUACUAGUACGUAAUUUG) is constructed and analyzed in three stages. In the first stage (B and C), long exact stems (series of positive numbers in the diagonals) are sought and classified. Next, in the second stage (D and E), for each positively classified exact stem, its diagonal is searched to form a non-exact stem (series of positive numbers interspersed with series of 0's) that also passes through a classification procedure. Finally, in the last stage (F and G), a complete hairpin is produced from each previously filtered non-exact stem, using the originally identified exact stem as the starting point for a further search in the matrix. In this search, other diagonals are tried so that secondary structures with asymmetrical internal loops are also considered. The resultant hairpins are then classified with a third ML model and only the ones predicted as positives are given as the final output (H).

Our main contribution here is the application of ML techniques with the objective of minimizing false positives. It contrasts with the verification of a list of criteria performed in miRNAFold. It is important to notice that each stage has its own ML model. For example, there is a specific model to apply on all exact stems

Algorithm 1 Construction of a triangular base pairing matrix

```
1: procedure BasePairingMatrix(s[0..n-1])
2: input: A putative pre-miRNA sequence of length n.
3: output: A triangular base pairing matrix.
4:     for i ← 0 to n-2 do
5:         if s[0] complements s[n-1-i] then
6:             M[i,0] ← 1
7:         else
8:             M[i,0] ← 0
9:         end if
10:        if s[n-1] complements s[i] then
11:            M[0,i] ← 1
12:        else
13:            M[0,i] ← 0
14:        end if
15:    end for
16:    for i ← 1 to n-3 do
17:        for j ← 1 to n-2-i do
18:            if s[n-1-i] complements s[j] then
19:                M[i,j] ← M[i-1,j-1]+1
20:            else
21:                M[i,j] ← 0
22:            end if
23:        end for
24:    end for
25:    return M
26: end procedure
```

of a minimum predefined length found in the first stage (Figure 1C). Only those instances considered as positives, i.e., for which the model assigns a probability value greater or equal to a predefined threshold, are given as input to the second stage. The non-exact stems produced from the positively classified exact stems, in turn, are classified with another ML model (Figure 1E), and only the instances regarded as positives pass to the next phase. In the last stage, a third ML model is used to classify (Figure 1G) the resultant hairpins to report the final predictions.

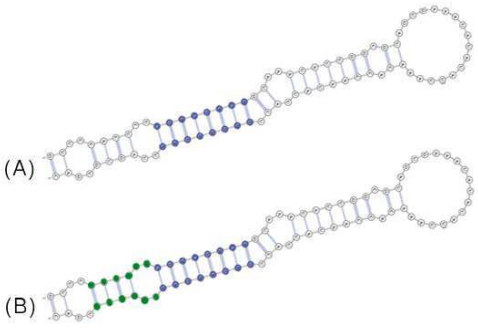

Figure 2 Illustration of the incremental approach performed in the base pairing matrix analysis. (A) A long exact stem (in blue) is identified (steps shown in Figure 1B/C). (B) The exact stem is then extended to a non-exact stem (Figure 1D/E, here in green and blue) that, in turn, is the basis to build a complete hairpin (procedure represented in Figure 1F/G).

The three-stages procedure described above is repeated for each sliding window subsequence. At the end of each analysis, the sliding window is moved 10 nt downstream, as proposed by Tempel and Tahi [14]. This process continues until the end of the given DNA sequence.

Datasets

To validate the proposed method, three datasets used in the experiments of miRNAFold were also used in our work to facilitate the comparisons. One of the datasets is an artificially constructed sequence obtained from 100 human pre-miRNAs interposed between human mRNAs, leading to a sequence of 30,500 nt. The other two datasets are real genomic data containing clusters of pre-miRNAs. The first one is a sequence extracted from the positive strand of the human chromosome 19, starting at position 54,169,933 and ending at position 54,485,651, containing 50 pre-miRNAs. The second sequence was obtained from the positive strand of the mouse chromosome 2, starting at position 10,388,290 and ending at position 10,439,906, comprehending 71 pre-miRNAs. These three datasets are referred throughout the text as HAD (human artificial data), HSD (*Homo sapiens* data), and MMD (*Mus musculus* data), respectively.

The sequences from zebrafish and sea squirt chromosomes used in the miRNAFold work were not included in our experiments because we could not match the stretches cited by the authors to the data available in GenBank. Furthermore, to our knowledge, there were no validated pre-miRNAs in the miRBase database for these two species at the moment we performed our experiments. Details about the datasets can be found in the work of Tempel and Tahi [14].

To build the learning models, three independent datasets to produce training sets (TSs) were constructed: TSHA, TSHS, and TSMM, one for each experiment with the three test sets: HAD, HSD, and MMD, respectively. Positive examples were obtained from experimentally validated pre-miRNA sequences present in miRBase release 21 [22]. For a fair model evaluation, all pre-miRNAs contained in the test sets were eliminated from TSHA, TSHS, and TSMM. Therefore, the positive instances in TSHA were a result of all validated human pre-miRNAs in miRBase subtracted by the pre-miRNAs present in HAD. Similarly, TSHS's positive instances were obtained after subtracting the pre-miRNAs in HSD from the set of validated human pre-miRNAs. Finally, the positive instances of TSMM were the result of the validated mouse pre-miRNAs in miRBase minus the pre-miRNAs in MMD. As a consequence, TSHA, TSHS, and TSMM contain 235, 295, and 364 positive examples, respectively.

Negative examples, in turn, were comprised of other types of non-coding RNAs along with pseudo hairpins. Gene sequences of snRNA (small nuclear RNA), snoRNA (small nucleolar RNA), tRNA, and miscRNA (miscellaneous RNA) were extracted from GenBank and the NRDR repository [26], totaling 2,480 sequences from the *H. sapiens* genome and 3,298 sequences from the *M. musculus* genome. Moreover, 1,872 pseudo pre-miRNAs from the work of Xue et al. were considered to compose the *H. sapiens* TSs [24]. Therefore, TSHA, TSHS, and TSMM contain 4,352, 4,352, and 3,298 negative examples, respectively.

Notice that each stage has its own TS to build its particular model. Therefore, each of the datasets TSHA, TSHS, and TSMM led to three TSs that are identified along the text with the indexes 1, 2, and 3, e.g., the dataset TSHA gave rise to the TSs: TSHA1, TSHA2, and TSHA3, one for each respective stage. Figure 3 illustrates this procedure. In this work, the TSs are in the ARFF format. This is the native format of the Weka machine learning toolkit [27], whose application programming interface (API) is used in the Mirnacle implementation.

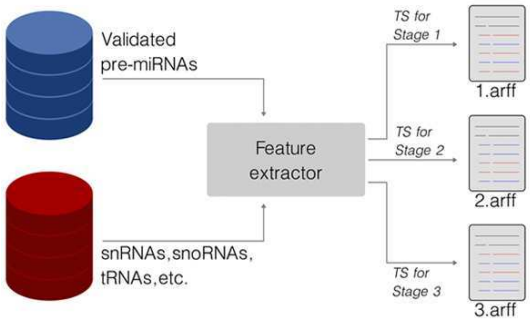

Figure 3 Training set construction. Each stage has its own training set for building its specific machine learning model. Real pre-miRNAs are used as positive examples, while negative examples are comprised of snRNAs, snoRNAs, tRNAs, miscRNAs, and pseudo hairpins.

Features

Another distinctive aspect of Mirnacle regarding miRNAFold was the inclusion of other features to improve the discriminatory power of the ML models. Besides the features proposed by Tempel and Tahi, we also considered dinucleotides, MFE (minimum free energy) 1, and MFE 2 from the work of Batuwita and Palade [28], as well as triplets from the work of Xue et al. [24]. Furthermore, the feature "size of the internal symmetric loop" was replace to other three features: Number of internal symmetric loops, average size of internal symmetric loops, and max size of internal symmetric loops. Still, the feature "size of the biggest side of the biggest bulge" was replaced to other two features: Size of the biggest bulge to the left of the terminal loop and size of the biggest bulge to the right of the terminal loop. Considering that dinucleotides and triplets mean 16 and 32 additional features, respectively, our approach includes 51 extra features. Notice that a triplet, according to Xue and colleagues [24], denotes whether each nucleotide in a consecutive sequence of three nucleotides is paired, represented by the symbol '(', or unpaired, represented by the symbol '.', and also specifies the middle nucleotide. Therefore, the feature "U((.", for example, means that there is a consecutive sequence of three nucleotides where the middle is U, the first and the second nucleotides are paired, and the third one is unpaired. The complete list of the features used in the ML model of each stage is given below:

- Features for exact stem classification in the first stage: Size, deltaG, percentage of each type of nucleotide, maximum number of consecutive nucleotides of each type, percentage of GU pairing, percentage of each possible dinucleotide, percentage of the difference between G+A and C+U, and percentage of the difference between G and C;
- Features for non-exact stem classification in the second stage: The same features of stage 1 plus percentage of base pairing, number of exact stems, average size of palindromes, number of internal symmetric loops, average size of internal symmetric loops, and maximum size of internal symmetric loops;
- Features for hairpin classification in the third stage: The same features of stage 2 plus average size of exact stems, size of the terminal loop, percentage

of GC pairing, adjusted MFE, percentage of non-exact stems covering the hairpin, size of the biggest difference between the two sides of bulge, size of the biggest bulge to the left of the terminal loop, size of the biggest bulge to the right of the terminal loop, difference between the number of left and right bulges, number of consecutive bulges, number of consecutive bulges on the same side, MFE1, MFE 2, and triplets.

The values of MFE and deltaG are obtained from the computational tools RNAfold and RNAcofold, respectively, which are part of the Vienna RNA package [16].

Learning with imbalanced datasets

As could be seen in the description of our data, the TSs are highly imbalanced. Namely, the number of negative instances is much greater than the number of positive instances. It may be a problem because the resulting model could be biased to the dominant class, presenting a poor accuracy to classify positive examples. In order to avoid this issue, three strategies to cope with imbalanced data were considered: Cost matrix, sampling, and the SMOTE filter [29, 30].

In cost matrices, it is possible to set the cost of misclassifying positive and negative instances. In our case, this cost is inversely proportional to the number of instances in the TS, i.e., the cost of misclassifying a positive instance, which belongs to the minority class, is much higher than the cost of misclassifying a negative instance. As a result, the weights of both classes in the training process are equalized.

Sampling and SMOTE were applied to alter the number of instances in the TS, such that the amount of instances in each class became even. To this end, sampling with replacement was used to undersampling the majority class instances, while keeping the minority class instances unchanged. The SMOTE technique, in turn, was used to oversampling the minority class elements, which eliminates the possibility of information loss. To achieve this, SMOTE combines the features of existing instances with the features of their nearest neighbors to create additional synthetic instances.

We selected a set of well known learning algorithms to be used in conjunction with the above methods for imbalanced data. Inspecting the literature about computational methods regarding learning tasks to miRNA discovery [19, 20, 31, 32], we have chosen the following learning algorithms to evaluate: SVM, multilayer perceptron (MLP), and random forest (RF). Our intention was to perform some experiments varying the combination of methods for imbalanced data with these learning algorithms to select the most appropriate approach. The results shown in the next section demonstrate that the best strategy among the possible combinations is SMOTE + RF.

To deploy all these ML methods in our computational tool, we used the API of the Weka machine learning toolkit [27], version 3.7.11, that implements all algorithms and techniques mentioned above. In particular for the SVM approach, we tested two implementations provided in the Weka API: Sequential minimal optimization (SMO) and LibSVM.

Complexity analysis

The analyses of time and space presented here do not consider the phase of ML model construction, because it is performed once and the resulting model is applied as much as needed.

Concerning space, given an input sequence of size n and a sliding window of size m, the space required by the algorithm as a function of these variables has complexity $\theta(n+m^2)$. Notice that it is necessary to store the whole sequence and the triangular base paring matrix. For large input sequences, such as an entire chromosome or even a whole genome, it is clear that $n \gg m$. In this case, we can consider the space complexity as $\theta(n)$.

Regarding time, for the same variables above, we can assume n window subsequences of size m to analyze, i.e., n executions of the three-stages pipeline, and an $m \times m$ matrix to explore in each case. Considering an extreme scenario in every stage where each of the m^2 entries of the matrix has to be processed, the matrix scanning has complexity $O(m^3)$, because for each position in the matrix, it is needed m additional entry accesses to process the respective diagonals. Therefore, the whole algorithm time complexity is $O(nm^3)$.

Results and discussion

Experiments were performed on a Linux machine, Intel Core Duo 2 T6600, 2.2 GHz, and 3 GB of RAM. Considering we used the same platform and processor reported in the miRNAFold work, we could directly compare our running times with the times the miRNAFold authors reported for their work and for other methods.

To evaluate the ML algorithms selected for classification as well as to compare our approach with other previously proposed methods, we report sensitivity (SN) and selectivity (SL) in our experiments, which are the same statistical measures used by Tempel and Tahi [14], and are given by:

$$SN = 100 \times \frac{TP}{(TP + FN)}, \qquad SL = 100 \times \frac{TP}{(TP + FP)},$$

where TP is the number of true positives, FN is the number of false negatives, and FP is the number of false positives. Furthermore, the geometric mean (GM) of SN and SL is provided in each experiment, as suggested by Gudyś et al. for the case of imbalanced data [32].

Deciding the ML methods to be part of Mirnacle

In order to decide the best method among the ones selected for treating the imbalance problem, all combinations of these methods with the already mentioned ML approaches were tested using TSHA1, TSHA2, and TSHA3. We kept the algorithms' parameters with their default values.

Table 1 shows the results of this experiment in which a 10-fold cross validation was performed for each possible combination. The best GM values for each stage and ML algorithm are underlined in the table. Compared with other methods for imbalanced data, the SMOTE technique presented the highest GM values in all stages regardless the ML algorithm used. Even analyzing SN and SL, individually, SMOTE demonstrates better results in most cases. Still in Table 1, it can be seen

Table 1 Comparing different combinations of methods for imbalanced data and learning algorithms. A 10-fold cross validation was performed in each case using TSHA1, TSHA2, and TSHA3 for the respective stages.

Method	1st stage			2nd stage			3rd stage		
	SN	SL	GM	SN	SL	GM	SN	SL	GM
				LibSVM:					
Cost matrix	4.7	64.7	17.4	3.8	81.8	17.7	4.2	100	20.6
Sampling	99.6	55.4	74.3	99.6	54.8	73.8	100	65	80.6
SMOTE	87.5	99.6	93.4	82.5	100	90.8	97.9	100	98.9
				SMO:					
Cost matrix	80.9	17.9	38	85.6	40	58.1	97.9	77.5	87.1
Sampling	84.3	77.7	81	86	90.6	88.3	98.7	96.7	97.7
SMOTE	86.3	83.5	84.9	91.9	94.4	93.1	99.9	99.3	99.6
				MLP:					
Cost matrix	74.1	16.7	35.2	79.2	49.7	62.7	98.7	3.9	19.6
Sampling	78.8	77.5	78.2	89.8	88.3	89.1	97	97.9	97.4
SMOTE	91.5	90.8	91.1	98	97.1	97.5	99.9	99.7	99.8
				RF:					
Cost matrix	46.2	44	45.1	74.6	76.5	75.5	89	89	89
Sampling	84.3	78.7	81.4	87.7	87.7	87.7	97.9	97.1	97.5
SMOTE	98.2	96.3	97.2	99.1	98.4	98.7	99.9	99.4	99.7

Table 2 Comparison of the selected classifiers combined to the SMOTE filter using TSHS1, TSHS2, and TSHS3 (A) as well as TSMM1, TSMM2, and TSMM3 (B) for the respective stages. Each ML algorithm was tested for each TS through a 10-fold cross validation.

Classifier	1st stage			2nd stage			3rd stage		
	SN	SL	GM	SN	SL	GM	SN	SL	GM
	(A) Results for TSHS1, TSHS2, and TSHS3:								
LibSVM	87.5	99.5	93.3	84.7	100	92	98.1	100	99.1
SMO	86.7	84	85.3	92.9	94.5	93.7	99.9	99.4	99.6
MLP	92.3	88.4	90.3	97.9	97	97.5	99.9	99.7	99.8
RF	97.8	97.7	97.8	98.7	99	98.8	100	99.7	99.8
	(B) Results for TSMM1, TSMM2, and TSMM3:								
LibSVM	89.4	98.8	94	100	60.5	77.8	97.6	100	98.8
SMO	85.6	81.6	83.5	92	92.5	92.2	99.2	99	99.1
MLP	88.9	86.8	87.8	95.2	94.9	95.1	99.9	99.5	99.7
RF	96.4	94.6	95.5	97.8	97.9	97.8	99.8	99.6	99.7

that the best GM values for stages 1, 2, and 3 with SMOTE were 97.2, 98.7, and 99.8, respectively, achieved by RF, RF, and MLP, in this order. Considering that the approach SMOTE+RF produced GM = 99.7 for the third stage, which is very close to the value achieved by SMOTE+MLP, the former demonstrated to be a good choice to be implemented in Mirnacle.

Further experiments with TSHS1, TSHS2, and TSHS3 (*Homo sapiens* TSs) as well as TSMM1, TSMM2, and TSMM3 (*Mus musculus* TSs) demonstrated the same. This time, we fixed SMOTE as the technique to cope with the imbalance problem, and varied the ML algorithm. Observing the best GM values underlined in Table 2, it can be noticed that the SMOTE+RF approach is, in fact, a suitable choice to integrate the Mirnacle method. RF achieved the highest values for GM in all stages for both organisms, i.e., it provided the best compromise between SN and SL. MLP also showed good results in the third stage. However, it presented worse GM values for the other phases compared with RF. Moreover, the training time with MLP is much longer than with RF (not shown).

Comparing Mirnacle with other methods for pre-miRNA ab initio prediction

After defining the best ML approach to use, we could conclude our computational tool and compare it with other previously proposed methods for pre-miRNA ab initio prediction. In our experiments, only ab initio methods of the third category, mentioned earlier, were considered.

Table 3 Comparison of the methods for pre-miRNA ab initio prediction using sequence HAD. The results of five distinct combinations of discriminant probabilities for the three respective stages are shown in parenthesis for Mirnacle.

Method	Sensitivity	Selectivity	GM	Time(mm:ss)
Mirnacle (0.3,0.3,0.7)	97	81.51	88.91	14:58
Mirnacle (0.4,0.4,0.7)	86	85.15	85.57	07:31
Mirnacle (0.5,0.5,0.7)	65	86.76	75.09	03:24
Mirnacle (0.6,0.6,0.7)	55	94.83	72.21	02:05
Mirnacle (0.8,0.8,0.7)	23	95.83	46.94	01:20
TVM	97	39.34	61.77	*
miRNAFold	97	19.17	43.12	00:0.84
miRPara	97	9.70	30.67	05:24
CID-miRNA	97	11.72	33.71	90:49
VMir	28	1.32	6.07	02:32

The Mirnacle parameters are: Minimum exact-stem size, sliding window size, the probability threshold of each of the three stages, minimum pre-miRNA size, and maximum pre-miRNA size. In all experiments, the minimum exact-stem size, the sliding window size, the minimum pre-miRNA size, and the maximum pre-miRNA size were set to 4, 150, 50, and 150, respectively.

Similarly to the experiments performed for miRNAFold, appropriate probability thresholds for the three stages were established using the artificially created dataset HAD. Notice that RF produces the probability of an example being positive, instead of a binary output. This is very useful because the discriminant probability can be used according to a particular goal. Hence, if sensitivity is more important, then a low threshold should be used. On the other hand, if selectivity is the priority, e.g., to minimize laboratory validations, then a high threshold is more appropriate. It is necessary to define the thresholds of each model used for each stage. For this end, we tried several combinations of thresholds (not shown) and selected five of them (represented by ordered triples) that led to different values for SN and SL: (0.3, 0.3, 0.7); (0.4, 0.4, 0.7); (0.5, 0.5, 0.7); (0.6, 0.6, 0.7); and (0.8, 0.8, 0.7). The triple (0.3, 0.3, 0.7) resulted in the best GM (Table 3) and was used to the comparisons made with datasets HSD and MMD (Table 4).

The results shown in Tables 3 and 4 for TVM were taken from its publication [23], while the results for miRNAFold, miRPara, CID-miRNA, and VMir were obtained from the work of Tempel and Tahi [14]. Using the same criterion as the authors of miRNAFold, a predicted pre-miRNA is considered true if the distance from its center to the center of the known hairpin is less or equal to 10% of the size of the latter.

It can be seen in Table 3 that the best GM value of Mirnacle was 88.91, representing an increase of approximately 44% regarding the best result among the other methods, namely, GM = 61.77 produced by TVM. Concerning the selectivity, which is the main bottleneck of the previously proposed approaches, Mirnacle could deliver a selectivity of 81.51 that means more than a two-fold increase compared with TVM. Notice that a high sensitivity value was kept by Mirnacle. In Table 3, the five results shown for different thresholds for the three stages can serve as a reference to threshold values to be used depending on a particular objective. If one is interested in high selectivity to reduce biological validations, the combination (0.8,0.8,0.7) is a good option, for instance.

Table 4 Comparison of the methods for pre-miRNA ab initio prediction using sequence HSD (A) and MMD (B). The results of Mirnacle are with thresholds (0.3, 0.3, 0.7) for the three stages, respectively.

Method	Sensitivity	Selectivity	GM
(A) Results for the *Homo sapiens* sequence (HSD):			
Mirnacle	100	29.52	54.33
TVM	100	1.43	11.95
miRNAFold	100	0.89	9.43
miRPara	98	0.93	9.54
CID-miRNA	38	0.69	5.12
VMir	100	0.56	7.48
(B) Results for the *Mus musculus* sequence (MMD):			
Mirnacle	98.61	47.33	68.31
TVM	100	4.13	20.32
miRNAFold	98.59	7.71	27.57
miRPara	98.59	5.34	22.94
CID-miRNA	29.58	0.82	4.92
VMir	88.73	2.93	16.12

Concerning the running time, Mirnacle's performance was highly variable. This is because low thresholds in the first and the second stages mean a less strict filter of exact and non-exact stems, i.e., the third stage will contain more sequences to extend to a complete hairpin, which leads to a longer running time. High thresholds in the first and second stages, on the other hand, mean fewer non-exact stems to expand at the end, speeding up the process. Notice that the matrix exploration in the third phase for inspecting different possibilities of a complete hairpin is the most computationally-expensive part. Comparing the Mirnacle execution that took 14 minutes and 58 seconds with the running time of other methods, Mirnacle could only overcome CID-miRNA. However, considering that we substantially improved selectivity, the time saved in laboratory experiments is likely more significant. It can be seen that the time taken by TVM was not reported. The reason for this is that the authors do not mention any experiment to measure time. Furthermore, TVM is available only as a webserver, which makes infeasible to measure its running time in a fair way.

Table 4 shows that Mirnacle provided even more significant improvements in selectivity for the other datasets. It can be seen an increase of 20-fold compared with TVM for sequence HSD (Table 4A), and an increase of 6-fold compared with miRNAFold for sequence MMD (Table 4B), while maintaining high sensitivity. The results for sequence HSD are particularly remarkable. For sequence MMD, Mirnacle missed one pre-miRNA, resulting in slightly lower sensitivity compared with TVM. However, the improvement in selectivity produced by Mirnacle led to a much higher GM value of this method.

Conclusions

In this work, we propose an extension of the miRNAFold method for pre-miRNA ab initio prediction to address the low selectivity issue of miRNAFold and other ab initio approaches. Our experiments have shown that our method, termed Mirnacle, substantially increased the selectivity compared with previously proposed procedures, whereas keeping high sensitivity.

To achieve these results, the main improvements implemented in Mirnacle were: The analysis of negative training examples, in addition to positive examples; the

use of machine learning techniques, namely, SMOTE combined with random forest; and the inclusion of other important hairpin features. Furthermore, Mirnacle allows the user to provide positive and negative examples to generate new models, which results in great flexibility in the use of our computational tool, i.e., as long as appropriate training examples are available, Mirnacle can be, in principle, used for other organisms of interest.

As a future work, we intend to improve Mirnacle's running time. First, the calculation of MFE and deltaG will be part of Mirnacle's code, in order to eliminate calls to the Vienna RNA package. Second, and most importantly, Mirnacle will implement parallel approaches, such as GPU (graphics processor unit), as the analyses of the subsequences in a genome are completely independent. Therefore, the parallelization of such analyses can certainly bring a huge gain in performance.

Addendum: URL for the computational tool
Mirnacle is available from: `http://www.labinfo.lncc.br/mirnacle`. The user can provide positive and negative examples to construct training sets, generate models, and search for new pre-miRNAs in a DNA or RNA sequence.

Competing interests
The authors declare that they have no competing interests.

Author's contributions
YBM, ATRV, and FRC designed all analyses; YBM and APO were responsible for carrying out the analyses; YBM and FRC wrote the initial draft of the manuscript; APO and ATRV contributed to posterior revisions to the final draft. All authors read and approved the final paper.

Acknowledgements
This research is supported by CAPES, CNPq, FAPEMIG, FAPERJ, and IFNMG.

Author details
[1] Department of Informatics, Universidade Federal de Viçosa, 36570-900 Viçosa, Brazil. [2] Instituto Federal do Norte de Minas, Rua Mocambi, 39800-430 Teófilo Otoni, Brazil. [3] Department of Computer Science, University of Sheffield, Western Bank S10 2TN, Sheffield, UK. [4] Laboratório Nacional de Computação Científica, Rua Getúlio Vargas 333, 25651-071 Petropólis, Brazil.

References

1. Ha, M., Kim, V.N.: Regulation of microRNA biogenesis. Nature reviews Molecular cell biology **15**(8), 509–524 (2014)
2. Carrington, J.C., Ambros, V.: Role of microRNAs in plant and animal development. Science **301**(5631), 336–338 (2003)
3. Suh, M.-R., Lee, Y., Kim, J.Y., Kim, S.-K., Moon, S.-H., Lee, J.Y., Cha, K.-Y., Chung, H.M., Yoon, H.S., Moon, S.Y., *et al.*: Human embryonic stem cells express a unique set of microRNAs. Developmental biology **270**(2), 488–498 (2004)
4. Williams, A.H., Liu, N., van Rooij, E., Olson, E.N.: MicroRNA control of muscle development and disease. Current opinion in cell biology **21**(3), 461–469 (2009)
5. van Rooij, E., Olson, E.N.: MicroRNA therapeutics for cardiovascular disease: Opportunities and obstacles. Nature reviews Drug discovery **11**(11), 860–872 (2012)
6. Wang, D., Zhang, Z., O'Loughlin, E., Wang, L., Fan, X., Lai, E.C., Yi, R.: MicroRNA-205 controls neonatal expansion of skin stem cells by modulating the PI(3)K pathway. Nature cell biology **15**(10), 1153–1163 (2013)
7. Shivdasani, R.A.: MicroRNAs: Regulators of gene expression and cell differentiation. Blood **108**(12), 3646–3653 (2006)
8. Ambros, V.: The functions of animal microRNAs. Nature **431**(7006), 350–355 (2004)
9. Poy, M.N., Eliasson, L., Krutzfeldt, J., Kuwajima, S., Ma, X., MacDonald, P.E., Pfeffer, S., Tuschl, T., Rajewsky, N., Rorsman, P., *et al.*: A pancreatic islet-specific microRNA regulates insulin secretion. Nature **432**(7014), 226–230 (2004)
10. Hilton, C., Neville, M., Karpe, F.: MicroRNAs in adipose tissue: Their role in adipogenesis and obesity. International Journal of Obesity **37**(3), 325–332 (2013)
11. Pereira, D.M., Rodrigues, P.M., Borralho, P.M., Rodrigues, C.M.: Delivering the promise of miRNA cancer therapeutics. Drug discovery today **18**(5), 282–289 (2013)
12. Terai, G., Okida, H., Asai, K., Mituyama, T.: Prediction of Conserved Precursors of miRNAs and Their Mature Forms by Integrating Position-Specific Structural Features. journal.pone.0044314 **7**(9), 11 (2012)
13. Hansen, T.B., Venø, M.T., Kjems, J., Damgaard, C.K.: miRdentify: High stringency miRNA predictor identifies several novel animal miRNAs. Nucleic acids research **42**(16), 124–124 (2014)
14. Tempel, S., Tahi, F.: A fast ab-initio method for predicting miRNA precursors in genomes. Nucleic acids research **40**(11), 80–80 (2012)
15. Grundhoff, A., Sullivan, C.S., Ganem, D.: A combined computational and microarray-based approach identifies novel microRNAs encoded by human gamma-herpesviruses. Rna **12**(5), 733–750 (2006)

16. Lorenz, R., Bernhart, S.H., zu Siederdissen, C.H., Tafer, H., Flamm, C., Stadler, P.F., Hofacker, I.L.: ViennaRNA package 2.0. Algorithms for Molecular Biology **6**, 26 (2011)
17. Tyagi, S., Vaz, C., Gupta, V., Bhatia, R., Maheshwari, S., Srinivasan, A., Bhattacharya, A.: CID-miRNA: A web server for prediction of novel miRNA precursors in human genome. Biochemical and biophysical research communications **372**(4), 831–834 (2008)
18. Joachims, T.: Advances in Kernel Methods, pp. 169–184. MIT Press, Cambridge, MA, USA (1999). Chap. Making Large-scale Support Vector Machine Learning Practical
19. Kumar, S., Ansari, F.A., Scaria, V.: Prediction of viral microRNA precursors based on human microRNA precursor sequence and structural features. Virology journal **6**(1), 129 (2009)
20. Wu, Y., Wei, B., Liu, H., Li, T., Rayner, S.: MiRPara: A SVM-based software tool for prediction of most probable microRNA coding regions in genome scale sequences. BMC bioinformatics **12**(1), 107 (2011)
21. Markham, N., Zuker, M.: UNAFold: Software for nucleic acid folding and hybridization. In: Keith, J. (ed.) Bioinformatics. Methods in Molecular Biology, vol. 453, pp. 3–31. Humana Press, Totowa, NJ (2008)
22. Kozomara, A., Griffiths-Jones, S.: miRBase: Annotating high confidence microRNAs using deep sequencing data. Nucleic acids research, 1181 (2014)
23. Titov, I.I., Vorozheykin, P.S.: Ab initio human miRNA and pre-miRNA prediction. Journal of bioinformatics and computational biology **11**(06) (2013)
24. Xue, C., Li, F., He, T., Liu, G.-P., Li, Y., Zhang, X.: Classification of real and pseudo microRNA precursors using local structure-sequence features and support vector machine. BMC bioinformatics **6**(1), 310 (2005)
25. Titov, I.I., Vorobiev, D.G., Ivanisenko, V.A., Kolchanov, N.A.: A fast genetic algorithm for rna secondary structure analysis. Russian chemical bulletin **51**(7), 1135–1144 (2002)
26. Paschoal, A.R., Maracaja-Coutinho, V., Setubal, J.C., Simões, Z.L.P., Verjovski-Almeida, S., Durham, A.M.: Non-coding transcription characterization and annotation. RNA biology **9**(3) (2012)
27. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: An update. ACM SIGKDD explorations newsletter **11**(1), 10–18 (2009)
28. Batuwita, R., Palade, V.: microPred: Effective classification of pre-miRNAs for human miRNA gene prediction. Bioinformatics **25**(8), 989–995 (2009)
29. Chawla, N.: Data mining for imbalanced datasets: An overview. In: Maimon, O., Rokach, L. (eds.) Data Mining and Knowledge Discovery Handbook, pp. 853–867. Springer, USA (2005)
30. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic minority over-sampling technique. arXiv preprint arXiv:1106.1813 (2011)
31. Ahmadi, H., Ahmadi, A., Azimzadeh-Jamalkandi, S., Shoorehdeli, M.A., Salehzadeh-Yazdi, A., Bidkhori, G., Masoudi-Nejad, A.: HomoTarget: A new algorithm for prediction of microRNA targets in Homo sapiens. Genomics **101**(2), 94–100 (2013)
32. Gudyś, A., Szcześniak, M.W., Sikora, M., Makałowska, I.: HuntMi: An efficient and taxon-specific approach in pre-miRNA identification. BMC bioinformatics **14**(1), 83 (2013)