BIOINFORMATION
Discovery at the interface of physical and biological sciences

open access

www.bioinformation.net

Software

Volume 8(12)

RAmiRNA: Software suite for generation of SVM-based prediction models of mature miRNAs

Vaibhav Tyagi & CVS Siva Prasad*

Bioinformatics & Applied Science Division, Indian Institute of Information Technology, Allahabad, India; CVS Siva Prasad – Email: shiva@iiita.ac.in; *Corresponding author

Received June 14, 2012; Accepted June 16, 2012; Published June 28, 2012

Abstract:
MicroRNAs (miRNAs) are short endogenous non-coding RNA molecules that regulate protein coding gene expression in animals, plants, fungi, algae and viruses through the RNA interference pathway. By virtue of their base complementarity, mature miRNAs stop the process of translation, thus acting as one of the important molecules in vivo. Attempts to predict precursor-miRNAs and mature miRNAs have been achieved in a significant number of model organisms but development of prediction models aiming at relatively less studied organisms are rare. In this work, we provide a suite of standalone softwares called RAmiRNA (RAdicalmiRNA detector), to solve the problem of custom development of prediction models for mature miRNAs using support vector machine (SVM) learning. RAmiRNA could be used to develop SVM based model for prediction of mature miRNAs in an organism or a group of organisms in a UNIX based local machine. Additionally RAmiRNA generates training accuracy for a quick estimation of prediction ability of generated model.

Availability: Usage manual and download link for RAmiRNA could be found at http://ircb.iiita.ac.in.

Background:

MicroRNAs (miRNAs) are post-transcriptional regulators that bind to complementary sequences on target messenger RNA transcripts (mRNAs), usually resulting in translational repression and gene silencing. By affecting gene regulation, miRNAs are likely to be involved in most biological processes, some as critical as insulin secretion, hematopoietic lineage differentiation and lipid metabolism **[1-3]**. Since experimental cloning methods for searching new miRNAs are less efficient, time consuming and very expensive, computational approaches are becoming more and more popular to choose miRNA candidates for further experimental validation. Thus, most computational methods utilize pre-miRNA sequences and/or their secondary structures to detect miRNAs or pre-miRNAs using support vector machines, random forest models and *ab initio* prediction models **[4-6]**. miRNAs arise from a precursor structure (pre-miRNA), a stem-loop structure having 80 nucleotides in its body, on average. This pre-miRNA is in turn derived out of a primary miRNA (pri-miRNA) which is a transcript of a miRNA gene. The different strategies successfully developed by few researchers for the prediction of pre-miRNAs are categorized largely as filter-based, machine learning, homology-based and target centered approaches **[7]**.

Here, we develop RAmiRNA - a toolbox for easy development of dynamic prediction models using support vector machine (SVM) learning. RAmiRNA uses an ordered pipeline of PERL scripts to extract and modify mature miRNA sequences from the miRBase database **[8]** and subsequently compute features for classification and prediction. RAmiRNA provides a straight and easy to use platform for making SVM-based models which can predict mature miRNAs.

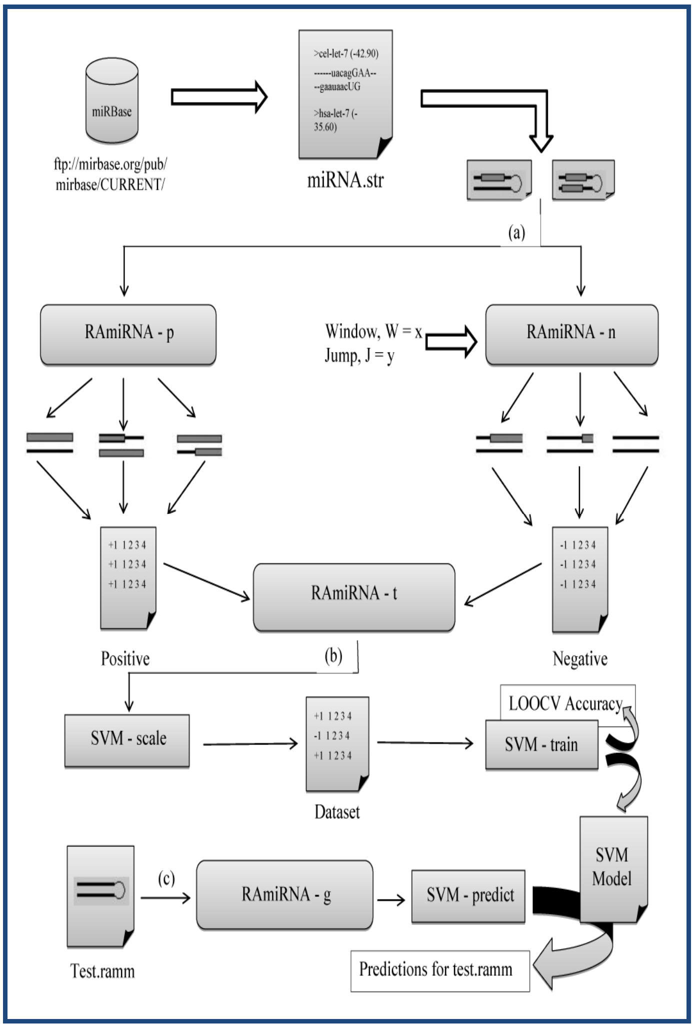

Figure 1: Flowchart illustrating the working pipeline of **RAmiRNA toolkit.** Part **(*a*)** of this figure demonstrates the ability of RAmiRNA – *p* and RAmiRNA – *n* to generate positive and negative datasets from a given miRBase organism id and miRNA.str database file. Note that RAmiRNA – *p* utilizes the standard miRBase format of writing a pre-miRNA to identify the mature miRNAs (shown here as boxes on a stem of pre-miRNAs). In part **(*b*),** working of RAmiRNA – *t* is shown. RAmiRNA – *t* combines the outputs of RAmiRNA – *p* and *n* to feed it into LibSVM's 'SVM-scale' and 'SVM-train' tools sequentially to generate a classification SVM model. It also reports cross validation accuracy. Finally, part **(*c*)** elucidates the process of testing a pre-miRNA (Test.ramm) using RAmiRNA – *g*.

Methodology:

RAmiRNA suite approaches the problem of mature miRNA prediction by using a sliding window protocol. Generally, in a sliding window approach to sequence analysis a virtual window of a particular length is placed over a linear sequence (of nucleotides/amino acids) from which meaningful score (or scores; number of nucleotides, for instance) is then calculated. In the next step, sliding window is shifted (we use the term 'jump' to denote this shift) by a few nucleotides and the score is calculated again. This procedure is repeated exhaustively.

RAmiRNA suite utilizes this protocol to implement a sliding window over secondary structures (stem loops) of pre-miRNAs to calculate a set of features. This is then fed into an SVM classifier. RAmiRNA suite builds the SVM based classifier on the basis of differentiation between the regions containing mature miRNA, with the region falling away from it.

RAmiRNA suite consists of four main tools: RAmiRNA-*p* generates positive set data which corresponds to the region of mature miRNA. RAmiRNA-*n* is used for negative set preparation which corresponds to region falling away from actual mature miRNA. RAmiRNA-*t* takes the two sets generated by RAmiRNA-*p*, and RAmiRNA-*n*, and combines these two sets into one (for details, see additional information provided in the supplementary). It then feeds this dataset into an efficient, publicly available support vector machine tool called LibSVM-train **[9],** which trains this dataset and generates the SVM prediction model. Finally, for actual testing of the pre-miRNAs, RAmiRNA-*g* generates test set and feeds it to LibSVM-predict, ultimately generating predictions in the form of graphical output showing mature miRNA regions. Work flow of RAmiRNA suite is illustrated in **(Figure 1).** RAmiRNA-*p* & RAmiRNA-*n* automatically labels the positive and negative entries respectively into typical LibSVM format. LibSVM tries to form a definite boundary between the two sets which ultimately serves as the basis of prediction for RAmiRNA-*g*.

Encoding features:

RAmiRNA utilizes some of the most basic, yet powerful features which broadly fall into two categories: sequence based features and structure based features. It encodes a set of forty-six useful features which are then selected on the basis of their statistically significant contribution towards training accuracy of the prediction model. **(Figure 2)** illustrates the significance of features used in RAmiRNA (see supplementary information for complete list of features).

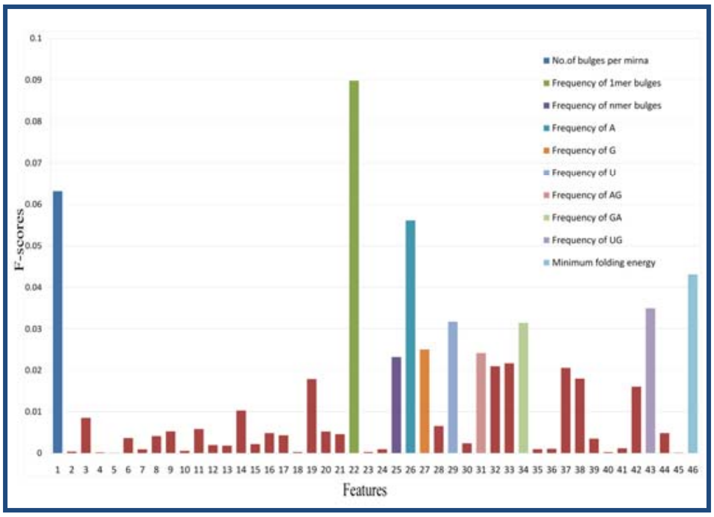

Figure 2: Statistical contribution of various features using F-scores. This bar graph illustrates the contribution of features used in RAmiRNA. Features with highest F-scores are color coded and listed in graph legend to differentiate them from relatively non-contributing features which are shown as red bars.

Input & Output:

In order to construct a classification model, RAmiRNA requires several inputs to be supplied. **a)** The complete miRBase database in the form of a downloadable text file (miRNA.str, see supplementary information for details); **b)** miRBase ID. For example, if a user wants to build a prediction model for viruses, then the ids to be supplied are ebv, hiv, bkv, rlcv etc. RAmiRNA-*p* and RAmiRNA-*n* utilize these inputs in a slightly different manner from each other. RAmiRNA-p extracts out the mature miRNA region from the pre-miRNA structures and encodes these structural entities into numerical values labeling them as +1. On the other hand, RAmiRNA-n traverses the stem of pre-miRNA structures by sliding a window of user defined length, avoiding the area containing mature miRNA, to encode numerical values which are labeled as -1. Consequently, RAmiRNA-*n* requires two more inputs: **c)** a window length, '*w*'; **d)** the jumps '*j*' that the window is expected to take upon the stem of pre-miRNA structures. Inputs to RAmiRNA-*t* are the outputs of RAmiRNA-*p* (Positive dataset) and RAmiRNA-*n* (Negative dataset). RAmiRNA-*t* generates a classification model as a result of training of the dataset. RAmiRNA-*t* also provides users with a training accuracy. This accuracy reflects the prediction reliability of the generated model. RAmiRNA-*g* needs this model as an input along with the window length and jump size same as those supplied to RAmiRNA-*n*. The tools that are included in RAmiRNA toolkit are an ordered set (or a pipeline) of Perl programming codes.

Caveat and future development:

Since RAmiRNA is dependent on number of miRNAs in miRBase database, some of the prediction models it generates are less accurate (for instance models for organisms having very few known miRNAs). Such models would become more reliable with the growth of miRBase in future. Some other classification features (such as enzyme recognition sites) would also be considered in future updates of RAmiRNA.

Acknowledgement:

Resources for the completion of this work were provided by Indian Institute of Information Technology, Allahabad. We would like to thank Atesh Koul & V.P.Subramanyam Rallabandi (National Brain Research Centre, Gurgaon) for giving useful suggestions.

References:

[1] Poy MN *et al. Nature*. 2004 **432**: 226 [PMID: 15538371]
[2] Chen CZ *et al. Science*. 2004 **303**: 83 [PMID: 14657504]
[3] Wilfred BR *et al. Mol. Genet. Metab.* 2007 **91:** 209 [PMID: 17521938]
[4] Xu, J-H *et al. Proteomics and Bioinformatics*. 2008 **6**: 121 [PMID: 18973868]
[5] Jiang P *et al. Nucleic Acids Res*. 2007 **35**: W339 [PMID: 17553836]
[6] Sewer A *et al. BMC Bioinformatics*. 2005 **6**: 267 [PMID: 16274478]
[7] Mendes ND *et al. Nucl Acid Res*. 2009 **37**: 2419 [PMID: 19295136]
[8] Griffiths-Jones S *et al. Nucl Acids Res*. 2008 **36***:* D154 [PMID: 17991681]
[9] http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Edited by P Kangueane
Citation: **Tyagi & Prasad,** Bioinformation 8(12): 581-585 (2012)

Supplementary material:

RAmiRNA is developed to work on Linux operating systems with both 32 bit and 64 bit CPU architecture. Basic requirements for proper functioning of this toolkit are summarized in the table below:

RAmiRNA toolkit	
Total number of tools/ executables	4 [RAmiRNA –p, RAmiRNA –n, RAmiRNA – t, RAmiRNA – g]
Operating system	Linux
CPU architecture	32 or 64 bit
Requirements	Perl, LibSVM standalone, Mfold standalone, GNUplot
Input	Complete miRBase database structure file

Perl
Perl is a programming language and is mostly included in various Linux distributions. Alternatively, it can be obtained from http://www.perl.org/get.html.

LibSVM
LibSVM can be downloaded from http://www.csie.ntu.edu.tw/~cjlin/libsvm/. Compiling LibSVM is then straightforward. A single 'make' command will automate the binary building process. It is important to note the complete path to the LibSVM directory. This path is used in execution of RAmiRNA – t and RAmiRNA –g. Some binaries in LibSVM are dependent on GNUplot which is readily available in most of the Linux software repositories.

Mfold
Standalone Mfold can be downloaded from http://mfold.rna.albany.edu/?q=mfold/download-mfold. Installation of Mfold requires C ++, C and Fortran compilers. After unpacking of the mfold download package, running the configure script in main mfold directory builds the 'make' file. This make file can then be used to call 'make install' command to install the mfold software.

miRBase
Complete miRBase database can be downloaded from ftp://mirbase.org/pub/mirbase/CURRENT/. miRNA.str file contains both miRNA sequences and secondary structures. Secondary structures of all the pre-miRNAs in miRBase are given in a characteristic four line text format. It is important to place the miRBase database structure file (miRNA.str) in the RAmiRNA directory.

RAmiRNA
(i) RAmiRNA – p is a tool to generate a positive dataset for a given organism's miRBase ID. For instance, to generate a positive dataset for Epstein Barr Virus, type –

```
./RAmiRNA – p  –id=ebv
```

It generates a text file containing positive dataset in LibSVM format (called pos_ebv for the above mentioned example).
(ii) RAmiRNA – n generates negative dataset for a given organism's miRBase ID. It can be called (for instance) as –

```
./RAmiRNA – n  -id=ebv  -w=22  -j=3
```

It generates another text file containing negative dataset in LibSVM format (called neg_ebv for the above mentioned example).
(iii) RAmiRNA – t takes a positive and a negative dataset to generate a binary classification model. It can be invoked by following command –

```
./RAmiRNA –t  -p=path_to_LibSVM
```

It is important to note here that RAmiRNA – t only accepts a positive dataset text file named as 'pos' and a negative dataset text file named as 'neg'. It in turn generates a classification model file called dataset.model inside the LibSVM directory.
(iv) RAmiRNA – g provides a graphical way to predict mature miRNAs in a given pre-miRNA structure. For instance, to test five pre-miRNAs for the presence of mature miRNAs each of the pre-miRNA structures should be converted to the default miRBase database format and placed in five different text files. The extensions of these text files should be changed to '.ramm'. Consequently, any file with a '.ramm' extension in RAmiRNA directory will be picked up by RAmiRNA – g for testing. The usage of RAmiRNA – g is –

```
./RAmiRNA – g  -w=22 –j=3 -p=path_to_LibSVM
```

List of features implemented in RAmiRNA sorted by their respective F-scores

Feature number	Name of the feature	Category	F-scores
22	Frequency of 1mer bulges	Structure	0.089831
1	Number of bulges per mirna	Structure	0.063173
26	Frequency of A in mirna	Sequence	0.056132
46	Minimum folding energy of part of hairpin having miRNA	Structure	0.04315
43	Frequency of UG in mirna	Sequence	0.034996
29	Frequency of U in mirna	Sequence	0.031759
34	Frequency of GA in mirna	Sequence	0.031479

27	Frequency of G in mirna	Sequence	0.025005
31	Frequency of AG in mirna	Sequence	0.024142
25	Frequency of nmer bulges	Structure	0.023168
33	Frequency of AU in mirna	Sequence	0.021675
32	Frequency of AC in mirna	Sequence	0.020958
37	Frequency of GU in mirna	Sequence	0.020565
38	Frequency of CA in mirna	Sequence	0.01797
19	Frequency of UG in bulges	Sequence/Structure	0.017868
42	Frequency of UA in mirna	Sequence	0.016033
14	Frequency of CA in bulges	Sequence/Structure	0.010262
3	Frequency of G in bulges	Sequence/Structure	0.008471
28	Frequency of C in mirna	Sequence	0.006539
11	Frequency of GG in bulges	Sequence/Structure	0.005805
9	Frequency of AU in bulges	Sequence/Structure	0.005241
20	Frequency of UC in bulges	Sequence/Structure	0.005204
16	Frequency of CC in bulges	Sequence/Structure	0.004843
44	Frequency of UC in mirna	Sequence	0.00484
21	Frequency of UU in bulges	Sequence/Structure	0.004571
17	Frequency of CU in bulges	Sequence/Structure	0.004308
8	Frequency of AC in bulges	Sequence/Structure	0.004119
6	Frequency of AA in bulges	Sequence/Structure	0.003632
39	Frequency of CG in mirna	Sequence	0.00349
30	Frequency of AA in mirna	Sequence	0.002346
15	Frequency of CG in bulges	Sequence/Structure	0.002179
12	Frequency of GC in bulges	Sequence/Structure	0.00195
13	Frequency of GU in bulges	Sequence/Structure	0.001804
41	Frequency of CU in mirna	Sequence	0.001157
36	Frequency of GC in mirna	Sequence	0.001017
35	Frequency of GG in mirna	Sequence	0.000952
24	Frequency of 3mer bulges	Structure	0.000939
7	Frequency of AG in bulges	Sequence/Structure	0.000913
10	Frequency of GA in bulges	Sequence/Structure	0.000537
2	Frequency of A in bulges	Sequence/Structure	0.000362
23	Frequency of 2mer bulges	Structure	0.000263
18	Frequency of UA in bulges	Sequence/Structure	0.000237
40	Frequency of CC in mirna	Sequence	0.000218
4	Frequency of C in bulges	Sequence/Structure	0.000172
45	Frequency of UU in mirna	Sequence	0.000088
5	Frequency of U in bulges	Sequence/Structure	0.000066