piTargetClassifier.py Manual

Yu Sun, 2018-12-13

1, Introduction

piTargetClassifier.py is a python program written in python3. It is a tool for piRNA-mRNA target classification based on complementary base pairing. This program requires numpy, sklearn and matplotlib packages.

A basic structure of the program:

- piTargetClassifier.py: Main program.
- Manual.pdf: Manual.
- bin: This folder contains scripts that are required by the main program. There are total 8 python scripts.

2, Modes

This program has multiple modes, and some modes can be run separately, but others cannot. This section provides a detailed explanation of each mode.

1, Show usage

Usage page will be shown if no parameters passed to the program, or if you specify the -h.

2, Demo mode

Without inputting any arguments, demo mode provides a test run using the data in /demo and pre-set parameters. The output files of Demo run are in /Results_Demo, and the log information has been saved as: /demo/demo.RunLog.txt. Apparently, Demo mode can be run separately. If you run this program in python2, or any required packages haven't been installed, Demo mode will fail.

3, Align mode

Align mode can take piRNA and mRNA FASTA files as inputs and perform *de novo* alignment. If you have pre-aligned results, they can also be imported by Align mode, without running the alignment process again. Align mode can be run separately, and the outputs are several text files and pdf figures:

- Text files:

AllAlignment: Results of piRNAs aligned to either Control or Target mRNAs, showing top 3 alignments. For example, the file names are: DemoRun.Control.AllAlignment.txt, DemoRun.Targets.AllAlignment.txt.

AllAlignmentPattern: A summary of piRNAs aligned to either Control or Target mRNAs, showing top 3 alignments. For example, the file names are: DemoRun.Control.AllAlignmentPattern.txt, DemoRun.Targets.AllAlignmentPattern.txt.

BestAlignment: Results of piRNAs aligned to either Control or Target mRNAs, showing only the best alignment. For example, the file names are: DemoRun.Control.BestAlignment.txt, DemoRun.Targets.BestAlignment.txt.

BestAlignmentPattern: A summary of piRNAs aligned to either Control or Target mRNAs, showing only the best alignment. For example, the file names are: DemoRun.Control.BestAlignmentPattern.txt, DemoRun.Targets.BestAlignmentPattern.txt.

- Figures:

WeightAve: Barplot for the average of weights at piRNA 1-30 positions. AllPattern uses original weight values to plot: For example: DemoRunPairedPlot.AllPattern.WeightAve.pdf, DemoRunPairedPlot.BestPattern.WeightAve.pdf. AllMatching uses the match/unmatch states to draw, which mean all matched weights will be 1, and unmatched positions are 0. This will eliminate the potential bias of nucleotides, only showing the matching status: DemoRunPairedPlot.AllMatching.WeightAve.pdf, DemoRunPairedPlot.BestMatching.WeightAve.pdf.

WeightSum: Barplot for the sum of weights, rather than average. For example: DemoRunPairedPlot.AllMatching.WeightSum.pdf, DemoRunPairedPlot.AllPattern.WeightSum.pdf, DemoRunPairedPlot.BestMatching.WeightSum.pdf, DemoRunPairedPlot.BestPattern.WeightSum.pdf

HistoPos: The histogram of the piRNA aligned to mRNAs, on a 1-100 nt scale. For example: DemoRunPairedPlot.AllPattern.HistoPos.pdf, DemoRunPairedPlot.BestPattern.HistoPos.pdf.

HistoScores: The histogram of scores (sum of each pattern pattern). For example: DemoRunPairedPlot.AllPattern.HistoScores.pdf, DemoRunPairedPlot.BestPattern.HistoScores.pdf

4, Learn mode

This mode contains 3 classifiers to learn patterns from piRNA:Target, piRNA: Control using a training dataset. After model fitting, it will evaluate the performance of these classifiers using test dataset. This mode cannot be run separately, and has to run with Align mode.

- Logistic classifier:

The output will be a pdf file (ROC curves on training and test datasets) and a txt log file. For example: DemoRun.ML.All.Logistic.CV5.pdf, DemoRun.ML.All.Logistic.CV5.txt. The default cross validation (CV) value is 5. If you change the CV value, the output file name will also change (CV + the CV number you are using).

- Random Forest classifier:

The output will be a pdf file (ROC curves on training and test datasets) and a txt log file. For example: DemoRun.ML.All.RandomForest.T100D8.pdf, DemoRun.ML.All.RandomForest.T100D8.txt, which means it used 100 trees and maximum depth 8.

- Support Vector Machine (SVM) classifier:

The output will be a pdf file (ROC curves on training and test datasets) and a txt log file. For example: DemoRun.ML.All.SVM.C1.pdf, DemoRun.ML.All.SVM.C1.txt, which means the penalty value has been set to 1.

5, Predict mode

This mode takes a pattern file (contains only one column of patterns) as input, and uses the classifier in the Learn mode to predict whether those patterns are from piRNA:Control (Control) or piRNA:Target (Targeted). This mode cannot be run separately. It must be run with Align and Learn modes.

3, Parameters

This section provides detailed usages of each parameters. For each mode, you need to specify --MODE (--Demo, --Align, --Learn or --Predict) first, and then specify required parameters. For Align mode, you need to choose either '*de novo*' sub-mode or 'import' sub-mode.

-h, --help	Show manual

Demo mode (--Demo)

Demo mode uses the prepared files in ./demo.
No other inputs needed. The test dataset will take ~1 min to run.

Align mode (--Align)

-v INT Verbosity, [1, 2, 3]. 1 will only a progress bar when running, and 3 will show all details. 2 will show a brief information.

Perform de novo alignments:

-p, -pi FILE piRNA FASTA file

-c, -mc FILE Control mRNA FASTA file

-t, -mt FILE Targeted mRNA FASTA file

-w Weight/FILE Specify 'match' or 'hy' will use default scoring systems (See Section 4) or you can specify an additional file, such as /demo/weight.hy.txt

-o Prefix Prefix of output files

or Use ready-to-use aligned results:

--import Prefix Prefix of pre-aligned data (All the .txt results)

Learn mode (--Learn)

--TestFrac Test data set fraction. For example, 0.3 means 30% data for testing.

-l, --logi Use logistic regression

--CVNum INT (Optional) Cross Validation N. Use default if not specified.

-r, --rf Use random forest classifier

--TNum INT (Optional) Tree Number N. Use default if not specified.

--Depth INT (Optional) Tree Depth N. Use default if not specified.

-s, --svm Use SVM classifier

--Cpen INT (Optional) SVM penalty C. Use default if not specified.

-a, --all Use all above three methods

-m, --mode MODE (Optional) All/Best/AllBest modes. Use default if not specified.

Predict mode (--Predict)

--prealign pre-aligned pattern file name.

--preout Prefix of the output files. For each classifier used in Learn mode, a preficted result file will be generated.

4, Default values

This section lists the default values for each parameter:

Parameters	Default values
-v	1
-w match	AT: 1, CG: 1, GG: 1, GT: 1
-w hy	AT: 2, CG: 3, GG: 1, GT: 2
--CVNum	5
--TNum	100
--Depth	8
--Cpenn	1
--mode	All
--TestFrac	0.05

5, Examples

Please make sure you have python3 installed on your computer, and it can be invoked in terminal. Also, the three required packages (numpy, sklearn, matplotlib) need to be installed. If you run this program on bluehive (at University of Rochester), you need to load the module: anaconda3 (module load anaconda3), which contains required python packages.

1, Show manual:

```
python3 piTargetClassifier.py
```

2, Run Demo mode:

```
python3 piTargetClassifier.py --Demo
```

Demo mode uses pre-set parameters which are same as the following command:

```
python3 piTargetClassifier.py --Align -v 1 -p ./demo/demo.piRNA.fa -
c ./demo/demo.Control.fa -t ./demo/demo.Target.fa -o DemoRun --
TestFrac 0.33 -w ./demo/weight.hy.txt --Learn -a -m AllBest --Predict
--prealign ./demo/demo.UnknownPatterns.txt --preout DemoPre
```

3, Run a real dataset, by performing *de novo* alignment (This will take ~15 h to finish. The SVM classifier cannot finish within 120 hours, so here I'm not running it):

```
python3 piTargetClassifier.py --Align -p Results_RealData/piRNA.fa -c
Results_RealData/RNA.Control.fa -t Results_RealData/RNA.Up.fa -w
demo/weight.hy.txt -o MyDataFinal --Learn -l -r -m AllBest
```

4, Run a real dataset, by importing pre-aligned results (for example, tune random forest classifier by changing the tree number and depth.):

```
python3 piTargetClassifier.py --Align --import MyDataFinal --Learn -r
--TNum 100 --Depth 15
```