PlncPRO User Manual

Contents

Essential requirements

- Operating System
  - Linux based
- Software requirements
  - Python 2.7
  - NCBI BLAST
  - framefinder(part of Estate package;provided with plncpro)
  - GNU C Library (glibc 2.12 or higher)
- Additional python modules
  - Regex
  - NumPy
  - SciPy
  - Biopython
  - Scikit-learn

*To install python packages we recommend to use pip

Setup

- Install Python 2.7 and the required modules
- Download and extract plncpro.tar.gz from here
- Make framefinder executable
  - Go to directory plncpro/lib/estate
  - Run sudo make
  - Copy/Move framefinder executable from plncpro/lib/estate/bin to plncpro/lib/framefinder
- Put the blast binaries in folder plncpro/lib/blast/bin
- Create a protein database using makeblastdb command to be used with blastx (swissprot recommended). e.g.:

  ```
  makeblastdb -in input_protein_file -title dbtitle -dbtype prot -out db_name -parse_seqids
  ```
- Run the required program from command line using python “script.py”

Usage and examples

1. **prediction.py**: To label lncRNAs and mRNAs. This file reads an input file containing sequences and then classifies the sequences as coding or non-coding. It uses a model generated by build.py to make classifications. It outputs a file containing class label and class probabilities for each sequence.

```
Usage: python prediction.py -i input_fasta_file -o output_directory -p
output_file_name -t number_of_threads -d path_to_blastdb -m
model_file
```

Parameters:

-p,--prediction_out	output file name
-i,--infile	input sequence file
-m,--model	model file
-o,--outdir	output directory name
-d,--db	path to blast database
Optional	
-t,--threads	number of threads [default: 4]
-l,--labels	path to the files containing labels(it outputs classification accuracy)
-r,--remove_temp	clean up intermediate files
-v,--verbose	show more messages on screen
--min_len	specifiy min_length to filter input files
--noblast	Don't use blast features
-no_ff	Don't use framefinder features
--qcov_hsp	specify query coverage parameter for blast [default:30]
--blastres	path to blast result for input file

Example

```
$ python prediction.py -i sample_data/test/neg.fa -p pred_res -o sample_preds -m
sample_out/sample_model -d lib/blastdb/sprotdb/sprotdb -t 10
```

Above command will label the sequences in the 'neg.fa' file using 10 threads. The output files will be written to the 'sample_preds' directory and 'pred_res' will contain the predicted class with probabilistic score. Each sequence predicted as mRNA will be labelled as 1 and lncRNAs will be labelled as 0.

2. **build.py**: used to build model using the given training data (mRNA/lncRNA transcripts). This file reads two labelled datasets containing coding and non-coding transcripts. Then it makes a random forest based classification model and saves the model, which can be used later to predict unknown sequences.

```
Usage: python build.py -p mRNAs_fasta -n lncRNAs_fasta -m
output_model_name -t number_of_threads -o output_dir -d
path_to_blast_database
```

Parameters:

-p,--pos	mRNA sequence file
-n,--neg	lncRNA sequence file
-m,--model	output model name
-o,--outdir	output directory name
-d	path to blast database
Optional	
-t,--threads	number of threads [default: 4]
-k,--num_trees	number of trees [default: 1000]
-r,--remove_temp	clean up intermediate files
-v,--verbose	show more messages
--min_len	specifiy min_length to use for prediction
--noblast	Don't use blast features
--no_ff	Don't use framefinder features
--qcov_hsp	specify query coverage parameter for blast [default:30]
--pos_blastres	path to blast result for mRNA input file
--neg_blastres	path to blast result for lncRNA input file

Example

a.)

```
$ python build.py -p sample_data/train/pos.fa -n
sample_data/train/neg.fa -o sample_out -m sample_model -d
lib/blastdb/sprotdb/sprotdb -t 10
```

NOTE: This constructs a model using the mRNA sequences in the 'pos.fa' file and lncRNA in 'neg.fa'. The program outputs the model in the file 'sample_model' in 'sample_out' directory. To use this model for predictions simply give the path to this model file as the -m,-- model argument in prediction.py, as below:

```
$ python prediction.py -i test.fa -out prediction_out -p prediction_file -m
sample_out/sample_model -d path_to_blast_db
```

b.)

```
$ python build.py -p sample_data/train/pos.fa -n sample_data/train/neg.fa -o sample_out -m sample_model -d lib/blastdb/sprotdb/sprotdb -t 10 --min_len 300
```

Above command will use all sequences from neg.fa and pos.fa having length greater than or equal to 300 bp for constructing the model.

3. **predtoseq.py**: used to extract mRNA or lncRNA sequences from PLNCPRO output file. This file reads a prediction output file and extracts sequences from a given class. User can specify class and probability cut-off and extract desired transcript sequences.

```
Usage: python predtoseq.py -f fasta_file -o outputfile -p PLNCPRO_prediction_file -l required_label -s 0.5
```

Parameters:

-f	input fasta file
-o	output fasta file name
-p	path to file containg predictions by PLNCPRO

Optional

-l	label of the required sequences (0 for lncRNA;1 for mRNA) [default:0]
-s	class probability cutoff (extract sequences with probability greater than or equal to s)
--min	specifiy min_length of sequences [default:0]
--max	specifiy min_length of sequences [default:Inf]

Example

```
$ python predstoseq.py -f fasta.fa -o output_lncRNA.fa -p PLNCPRO_pridiction_file -s 0.5
```

Above command will extract the lncRNA sequences having coding probability of less 0.5 predicted from PLNCPRO in the file output_lncRNA.fa.

Description of files

a. build.py: this file reads two labelled datasets containing coding and non-coding transcripts. Then it makes a random forest based classification model and saves the model, which can be used later to predict unknown sequences.

b. prediction.py: this file reads an input file containing sequences and then classifies the sequences as coding or non-coding. It uses a model generated by build.py to make classifications. It outputs a file containing class label and class probabilities for each sequence.

c. predtoseq.py: this file reads a prediction output file and extracts sequences from a given class. User can specify class and probability cut-off and extract desired transcript sequences.

d. blastparse.py: this file reads output of blastx program, run with “-outfmt '6 qseqid sseqid pident evalue qcovs qcovhsp score bitscore qframe sframe”, and extracts features from it.

e. extractfeatures.py: this file extracts trimer frequency and lengths from input fasta sequence.

f. ffparse.py: this file reads output from framefinder and extract features.

g. mergefeatures.py: this file merges all the features generated from blastpare.py, extractfeatures.py and ffparse.py in to single feature file.

h. buildmodel.py: this file reads an input file containing features and labels and outputs a random forest classification model

i. **predict.py**: this file reads an input feature file and predicts its label using a model.

Contact

- Urminder Singh (urmind13_sit@jnu.ac.in)
- Dr. Mukesh Jain (mjain@jnu.ac.in)

Terms of Use

PLNCPRO is free and open source software published under GNU Public License version 3. In no event will SCIS, JNU be liable to you for damage, including any general, special, consequential or incidental damage arising out of the use, modification or inability to use the program (including but not limited to loss of data or data being rendered inaccurate or losses sustained by you or third parties or a failure of the program to operate with any other programs).

THIS PACKAGE IS PROVIDED “AS IS” AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Use of this software is taken as an agreement to these terms of usage.