

PLNCPRO User Manual

Contents

Essential requirements	2
Setup	2
Usage and examples.....	3
Description of files	6
Contact.....	7
Citation	7
Terms of Use	8

Essential requirements

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

Setup

- Install Python 2.7 and the required modules
- Download and extract plncpro.1.0.tar.gz from [here](#)
- Make *framefinder* executable
 - Go to directory plncpro/lib/estate
 - \$ sudo make
 - Copy *framefinder* executable to plncpro/lib/framefinder
 - \$ cp bin/framefinder ../framefinder
- Put the blast binaries in folder plncpro/lib/blast/bin
- Create a protein database to be used with blastx (swissprot recommended)
- 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 specify 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

Examples:

a.) \$ 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.

b.) \$ 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 --min_len 500

This command is same as above but it will use sequences with length greater than or equal to 500 bp for prediction.

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

<code>--min_len</code>	specifiy min_length to use for prediction
<code>--noblast</code>	Don't use blast features
<code>--no_ff</code>	Don't use framefinder features
<code>--qcov_hsp</code>	specify query coverage parameter for blast
[default:30]	
<code>--pos_blastres</code>	path to blast result for mRNA input file
<code>--neg_blastres</code>	path to blast result for lncRNA input file

Examples:

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`

This 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`

Parameters:

-f	input fasta file
-o	output fasta file name
-p	path to file containing 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	specify min_length of sequences [default:0]
--max	specify min_length of sequences [default:Inf]

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. ffpars.py: this file reads output from framefinder and extract features.
- g. mergefeatures.py: this file merges all the features generated from blastparse.py, extractfeatures.py and ffpars.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

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

Citation

Please cite this article if used for any publication:

Singh, U., Khemka, N., Rajkumar, M. S., Garg, R., & Jain, M. (2017). PLncPRO for prediction of long non-coding RNAs (lncRNAs) in plants and its application for discovery of abiotic stress-responsive lncRNAs in rice and chickpea. *Nucleic Acids Research*, 45(22), e183. <https://doi.org/10.1093/nar/gkx866>

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 MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

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