The User Manual of PSM Validator

Long WU
HKUST

2021 March

Contents

1 Introduction

Validate PSM with a universal score.

2 Installation

2.1 Install with binary package

It is highly recommended to use the compiled exectuables due to the difficulty to compile C++ source code and reconcile all the dependencies. Download from here

Table 1: dependencies of PSMValidator

package	version	provided	function
spdlog	x	YES	logger
comet	x	YES	generate ground truth PSMs
ranger	x	YES	random forest
liblinear	2.21	YES	Logistic Regression
rapidxml	1.13	YES	xml parser
SpectraST	5.0	YES	spectra annotation
sqlite3	3.0	YES	
Boost	1.65	YES	File system; program option parser
gnuplot-iostream	x	YES	generate plots
gnuplot	x	NO	needed by gnuplot-iostream
gsl	1.16	NO	SpectraST

Those are the binary files in the bin folder:

```
.
|-- bin
|   |-- comet
|   |-- predict
|   |-- psmvalidator
|   |-- ranger
|   `-- train
```

2.2 Compile from source code

2.2.1 Prerequisites

The following packages are required to compile from source code
The following script installs the prerequisites packages.

Listing 1: install GSL v1.16 and other dependencies

```
# part one, install GSL 1.16
GSL_VERSION=$(gsl-config --version)
if [ "$GSL_VERSION" == "1.16" ]; then
```

```
  echo "GSL version 1.16 already installed!"
else
  echo GSL version is $GSL_VERSION
  wget https://mirror-hk.koddos.net/gnu/gsl/gsl-1.16.tar.gz
  tar xvf gsl-1.16.tar.gz
  # shellcheck disable=SC2164
  (
    cd gsl-1.16 || exit
    ./configure
    make all -j 10
    sudo make install
  )
fi

# part two install other packages.
sudo apt-get install sqlite3 libsqlite3-dev gnuplot gnuplot-
   qt
```

2.2.2 Compilation and installation from source code (Ubuntu)

First, install cmake, build-essential and gnuplot with following commands

```
sudo apt-get install cmake build-essential gnuplot
```

Next, compile and install the psmvalidator with following commands.

Listing 2: compile and install from soruce code of PSMValidator

```
unzip PSMValidator_v1.0.0.zip
cd </path/to/the/source/code>  # change accordingly
cmake .
cmake --build build
cmake --install . --prefix .
```

3 Basic usage

3.1 Tree structure after installed

```
.
|-- bin
|   |-- comet
|   |-- predict
|   |-- psmvalidator
|   |-- ranger
|   `-- train
|-- model
|   |-- nist_human_hcd_selected_features.txt_2.000000_frag.
   model
|   `-- nist_human_hcd_selected_features.
   txtmtry_8_ntree_900_trN_20000.forest
|-- param
|   `-- psmvalidator.conf
|-- predict
|   |-- 01625b_GD2-TUM_first_pool_12_01_01-3xHCD-1h-R1.mzML
|   |-- interact-01625b_GD2-TUM_first_pool_12_01_01-3xHCD-1h-
   R1.ipro.pep.xml
|   `-- TUM_first_pool_12.pep.list
|-- scripts
|   |-- train_small_demo_mgf.bash
|   `-- validate_psm.bash
`-- train
    |-- comet16low.param
    |-- featurelist.txt
    |-- features.txt
    |-- human_hcd_selected_new_small.mgf
    |-- uniprot-human-2020-12.fasta
    |-- uniprot-human-2020-12TD_only_decoy.fasta
    |-- uniprot_yeast_reviewed_6721_Nov152016.fasta
```

```
`-- uniprot_yeast_reviewed_6721_Nov152016TD_only_decoy.
    fasta
```

3.2 Train a demo validator

- input
  - spectra (mgf)
    - `train/human_hcd_selected_new_small.mgf`
  - database (fasta)
    - target: `uniprot-human-2020-12.fasta`
    - decoy: `uniprot-human-2020-12TD_only_decoy.fasta`
- output
  - model
    - `human_hcd_selected_new_small_mtry_8_ntree_900_features.forest`

to train the validator, run the following command.

```
cd scripts
./train_small_demo_mgf.bash
```

3.3 Prediction

- input
  - Search result: Comet search result (PepXML)
  - Model: `human_hcd_selected_new_small_mtry_8_ntree_900_features.forest`
- output
  - result table with RF(random forest) score

To use the validator, users could directly use the trained model. Here is an example:

```
cd scripts
./validate_psm.bash
```