iScore Documentation

Nicolas Renaud

Jun 25, 2022

Contents:

CHAPTER 1

Introduction

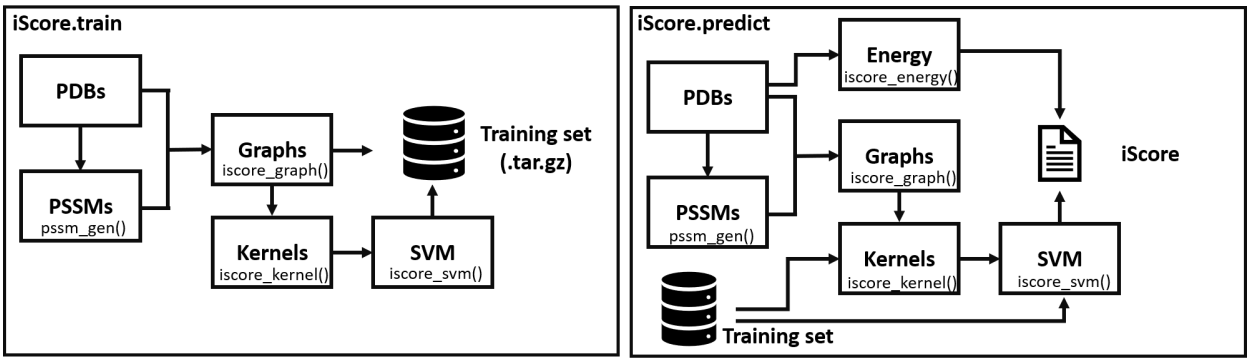
Support Vector Machine on Graph Kernels for Protein-Protein Docking Scoring

The software supports the publication of the following articles:

C. Geng *et al.*, *iScore: A novel graph kernel-based function for scoring protein-protein docking models*, bioRxiv 2018, https://doi.org/10.1101/498584

iScore uses a support vector machine (SVM) approach to rank protein-protein interfaces. Each interface is represented by a connection graph in which each node represents a contact residue and each edge the connection between two contact residues of different proterin chain. As feature, the node contains the Position Specific Similarity Matrix (PSSM) of the corresponding residue.

To measure the similarity between two graphs, iScore use a random walk graph kernel (RWGK) approach. These RWGKs are then used as input of the SVM model to either train the model on a training set or use a pretrained model to rank new protein-protein interface.



CHAPTER 2

Installation

2.1 Install via pip

In most cases you can use pip to install iScore. Simply use:

pip install iScore

2.2 Install from source

If the pip install fails or if you want to modify the code you can install iScore manually. The code is hosted on Github (https://github.com/DeepRank/iScore)

To install the code

- clone the repository `git clone https://github.com/DeepRank/iScore.git`
- go there `cd iScore`
- install the module `pip install -e ./`

To test the module go to the test folder `cd ./test` and execute the following test : `pytest`

CHAPTER 3

iScore Workflow

One of the mainfeature of the software are the serial and MPI binaries that fully automatize the workflow and that can be used directly from the command line. To illustrate the use of these binaries go to the folder `iScore/example/training_set/`. This folder contains the subfolders `pdb/` and `pssm/` that contain the PDB and PSSM files of our training set. The binary class corresponding to these PDBs are specified in the file 'caseID.lst'.

Training a model using iScore can be done in a single line using MPI binaries with the command :

```
$ mpiexec -n 2 iScore.train.mpi
```

This command will first generate the graphs of the conformations stored in `pdb/` using the PSSM contained in `pssm/` as features. These graphs will be stored as pickle file in `graph/`. The command will then compute the pairwise kernels of these graphs and store the kernel files in `kernel/`. Finally it will train a SVM model using the kernel files and the `caseID.lst` file that contains the binary class of the model.

The calculated graphs and the svm model are stored in a single tar file called here `training_set.tar.gz`. This file contains all the information needed to predict binary classes of a test set using the trained model.

To predict binary classes (and decision values) of new conformations go to the subfoler `test/`. Here 5 conformations are specified by the PDB and PSSM files stored in `pdb/` and `pssm/` that we want to use as a test set. Ranking these conformations can be done in a single command using :

```
$ mpiexec -n 2 iScore.predict.mpi --archive ../training_set.tar.gz
```

This command will use first compute the graph of the comformation in the test set and store them in *graph/*. The binary will then compute the pair wise kernels of each graph in the test set with all the graph contained in the training set that are stored in the tar file. These kernels will be stored in `kernel/`. Finally the binary will use the trained SVM model contained in the tar file to predict the binary class and decision value of the conformations in the test set. The results are then stored in a text file and a pickle file `iScorePredict.pkl` and `iScorePredict.txt`. Opening the text file you will see :

Name	label	pred	decision_value
1ACB_2w	None	0	-0.994
1ACB_3w	None	0	-0.994
1ACB_1w	None	0	-0.994
1ACB_4w	None	0	-0.994
1ACB_5w	None	0	-0.994

The ground truth label are here all None because they were not provided in the test set. This can simply be done by adding a `caseID.lst` in the `test/` subfolder.

3.1 Serial Binaries

Serial binaries are also provided and can be used in a similar way than the MPI binaries : `iscore.train` and `iscore.predict`

CHAPTER 4

Computing PSSM files

As a prepocessign step one must compute the PSSM files corespondng to the PDB files in the training/testing dataset. Thiscan be acheived with the PisBLast library (https://ncbiinsights.ncbi.nlm.nih.gov/2017/10/27/blast-2-7-1-now-available/). The library BioPython allows ane asy use of these libraries.

iScore contains wrapper that allows to compute the PSSM data, map them to the PDB files and format them for further processing. The only input needed is the PDB file of the decoy. To compute the PSSM file one can simply use :

```
>>> from iscore.pssm.pssm import PSSM
>>>
>>> gen = PSSM('1AK4')
>>>
>>> # generates the FASTA query
>>> gen.get_fasta()
>>>
>>> # configure the generator
>>> gen.configure(blast=<path to blast binary>, database=<path to the blast db>)
>>>
>>> # generates the PSSM
>>> gen.get_pssm()
>>>
>>> # map the pssm to the pdb
>>> gen.map_pssm()
```

CHAPTER 5

Graphs and Kernels

5.1 Generating the Graphs :

The first step in iSCore is to generate the connections graph of the itnerface. In this graph each node is represented by the PSSM of a residue. The nodes are connected if they form a contact pair between the two proteins.

To create the graph one needs the PDB file of the interface and the two PSSM files (one for each chain) created by the PSSMGen tool. To generate the graph simply use :

```
>>> from iScore.graph import GenGraph, Graph
>>>
>>> pdb = name.pdb
>>> pssm = {'A':'name.A.pdb.pssm','B':'name.B.pdb.pssm'}
>>>
>>> g = GenGraph(pdb,pssm)
>>> g.construct_graph()
>>> g.export_graph('name.pkl')
```

This simple example will construct the connection graph and export it in a pickle file. A working example can be found in `example/graph/create_graph.py`

The function `iscore_graph()` facilitate the generation of a large number of conformations. By default this function will create the graphs of all the conformations stored in the subfolder `./pdb/` using the pssm files stored in the subfolder `./pssm/`. The resulting graphs will be stored in the subfolder `./graph/`.

5.2 Generating the Graph Kernels :

Once we have calculated the graphs of multiple conformation we can simply compute the kernel of the different pairs using iScore. An example can be found at `example/kernel/create_kernel.py`

```
>>> from iScore.graph import Graph, iscore_graph
>>> from iScore.kernel import Kernel
```

(continues on next page)

(continued from previous page)

```
>>>
>>> # create all the graphs of the pdb in ./pdb/
>>> iscore_graph()
>>>
>>> #init and load the data
>>> ker = Kernel()
>>> ker.import_from_mat()
>>>
>>> # run the calculations
>>> ker.run(lamb=1.0,walk=4,check=checkfile)
```

The kernel between the two graphs computed above is calculated with the class *Kernel()*. By default the method *Kernel.import_from_mat()* will read all the graphs stored in the subfolder *graph/*. To compute all the pairwise kernels of the graphs loaded above we can simply use the method *Kernel.run()*. We can here specify the value of lambda and the length of the walk.

CHAPTER 6

Visualizing the connection graphs

iSore allows to easily visualize the connection graphs using the HDF5 browser provided with the software and pymol. First the connections graphs must be stored in a HDF5 file. To do that simply generate the graphs as following:

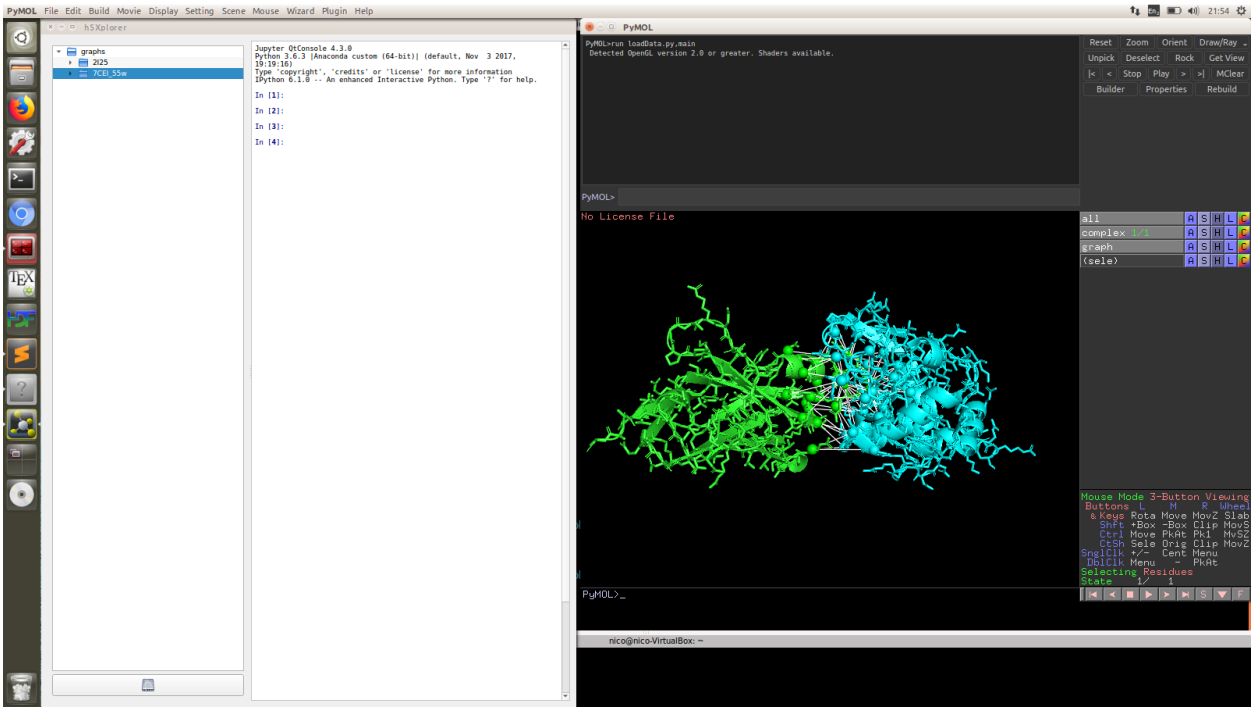
```
>>> from iScore.graphrank.graph import iscore_graph
>>> iscore_graph(pdb_path=<pdb_path>,
>>>              pssm_path=<pssm_path>,
>>>              export_hdf5=True)
```

where you have to specify the folder containing the PDB files abd PSSM files in pdb_path and pssm_path. By default this are simply `./pdb/` and `./pssm/`. The script above will create a HDF5 file containing the graph.

This HDF5 cile can be explored using the the dedicated HDF5 browser. Go to the `./h5x/` folder and type:

```
./h5x.py
```

This will open the hdf5 browser. You can open a hdf5 file by clicking on the file icon in the bottom left of the browser. Once opened, you will see the content of the file in the browser. Right-click on the name of a conformation and choose `3D Plot`. This will open PyMol and allow you to visualize the connecton graph


PyMOL File Edit Build Movie Display Setting Scene Mouse Wizard Plugin Help
h5Xplorer
graphs
2025
7CEI_55w
Jupyter QtConsole 4.3.0
Python 3.6.3 |Anaconda custom (64-bit)| (default, Nov 3 2017, 19:19:16)
Type 'copyright', 'credits' or 'license' for more information
IPython 6.1.0 -- An enhanced Interactive Python. Type '?' for help.
In [1]:
In [2]:
In [3]:
In [4]:
PyMOL
PyMOL>run loadData.py,main
Detected OpenGL version 2.0 or greater. Shaders available.
Reset Zoom Orient Draw/Ray
Unpick Deselect Rock Get View
Stop Play MClear
Builder Properties Rebuild
PyMOL>
No License File
all
complex 1:1
graph
(sele)
Mouse Mode 3-Button Viewing
Buttons L M R Wheel
Selecting Residues
State 1/ 1
PyMOL>_
nico@nico-VirtualBox: ~

CHAPTER 7

Documentation

All the prototypes of the class/methods are here specified

7.1 Graph

7.2 Kernel

7.3 SVM

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Python Module Index

i

Index

I