TUTORIAL
wisefrog

JONE GUTIÉRREZ DÍEZ
MARTÍN SUÁREZ ÁLVAREZ

MSc Bioinformatics for Health Science
Universitat Pompeu Fabra

INTRODUCTION

wisefrog is a Deep Learning based model that classifies ligand binding sites (LBS) according to the physical and chemical properties of atoms. It provides a characterization of atoms by its environment (geometrically, the atoms that surround it), in order to get a precise and specific perspective of how LBS are composed and classifies whether an atom or residue belongs to a LBS or not. Thus, the main target of this software is bioinformaticians and researchers who study structural biology, but can be used by anyone by following this tutorial.

As this tutorial focuses on a Python-based model, it will comprehensively cover its usage, starting from basic operations such as program initialization and simple output generation, all the way to exploring its full potential in handling more complex tasks and functionalities.

GETTING STARTED

SETTING UP THE ENVIRONMENT: INSTALLATION

The first step to follow is downloading the setup package from the GitHub repository (in the dist folder).

A specific version (if necessary) of all the required packages will be downloaded when installing wisefrog. Thus, we recommend creating a new Conda environment to avoid incompatibilities.

```
conda create wisefrog_env
conda activate  wisefrog_env
```

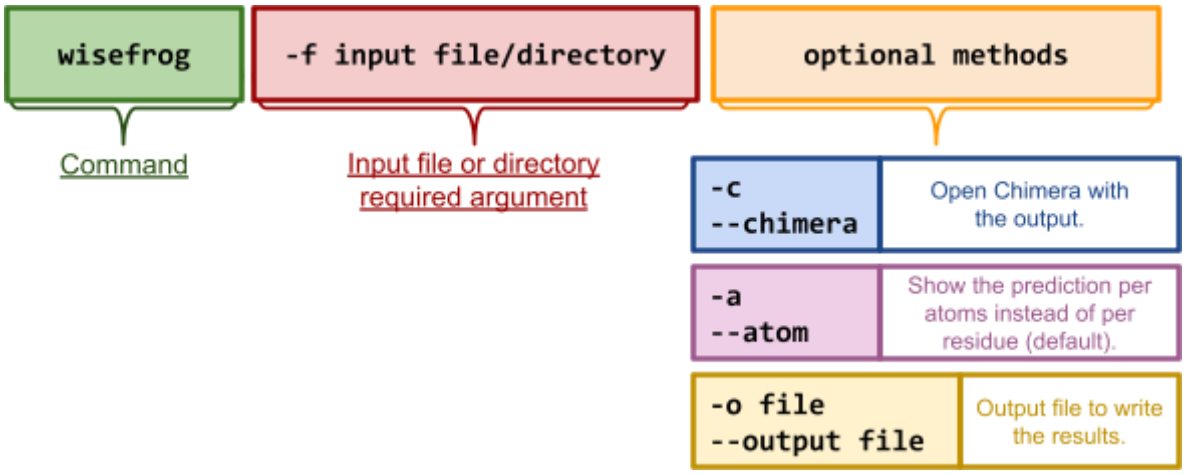
We'll proceed to the installation:

```
pip install wisefrog-0.1.tar.gz
```

This might take some minutes, be patient! Once the installation has ended the user will be able to predict ligand binding sites with wisefrog.

BASIC CONCEPTS

This package gives priority to user-friendliness and ease of use. So, always remember following the basic structure we suggest:



Bear in mind that all the flags are compatible with each other and, also, that either of the flag names can be used (long and short version).

WALKTHROUGH

USING WISEFROG TO PREDICT LBS FROM A SINGLE PDB FILE

We'll start running a simple command, wisefrog requires the file flag in order to run properly – note that wisefrog will only read files in pdb format:

```
wisefrog -f file_name.pdb
```

The output will be displayed in your terminal following this structure:

```
========File: file_name.pdb========
RESIDUE:
# List of residues.
```

USING WISEFROG TO PREDICT LBS FROM A PDB FILE FROM RCSB PDB

It is just the same as before but in this case wisefrog will get the PDB file directly from RCSB PDB, so the user doesn't have to download it. This will download the PDB file, predict the LBS and immediately remove the file:

```
wisefrog -f web/pdb_id
```

In this case wisefrog will directly request the file, and the output will be displayed in your terminal following this structure:

```
Downloading from https://files.rcsb.org/download/pdb_id.pdb
========File: pdb_id.pdb========
RESIDUE:
# List of residues.
```

USING WISEFROG TO PREDICT LBS FROM MULTIPLE PDB FILES

To analyze multiple PDB files at the same time the only requirement is to store all the files in the same directory, then run this command line:

```
wisefrog -f directory_name
```

The output will be displayed in your terminal following this structure:

```
========File: directory_name/file_name_1.pdb========
RESIDUE:
# List of residues.

========File: directory_name/file_name.pdb========
RESIDUE:
# List of residues.
```

VISUALIZATION OF THE RESULT

In order to get a visual support of the prediction, wisefrog reckons with a method to open Chimera:

```
wisefrog -f file_name -c
```

This time, the output will also be displayed in the terminal, but additionally Chimera will be opened, displaying the residues classified as LBS in red.

In case the user is working with several files, they will be shown in Chimera one by one after the prediction of each of them is done.

Note: In the examples' section the chimera output will be shown.

GETTING THE OUTPUT PER ATOM

In case the user desires to get a more specific output, wisefrog contains a method for it, which can be used together with all other methods:

```
wisefrog -f file_name.pdb -c -a
```

The output will be shown in the terminal as following:

```
========File: file_name.pdb========
ATOM         RESIDUE
# List of atoms and residues organized in columns as titles refer.
```

REDIRECTING THE OUTPUT TO AN OUTPUT FILE

Saving the results is crucial for a scientist, thus output flag will manage this task:

```
wisefrog -f file_name.pdb -c -a -o output.out
```

In this case the output won't be shown in the terminal, but in the desired output file. Take into account that if the output file is not empty, the content won't be overwritten by the output, but appended. The format of the output will be the same as the one shown when displayed in the terminal.

Note that in bash, flags can be combined, as in `-cao`.

EXAMPLES AND USE CASES

As mentioned during this tutorial and the documentation, this software is meant for predicting LBS, which can be the first step for many research projects involved in drug discovery and design, and many other fields of life sciences.

This section aims to introduce some examples of use of wisefrog, in order the user gets more used and comfortable with the commands and outputs. The PDB ID used as reference will be 1N07, and we'll use the bHLH family as the directory of PDBs.

First of all, to follow this tutorial we suggest creating a directory in order to store the files generated with the provided examples.

```
mkdir WF_tutorial
cd WF_tutorial
```

EXAMPLE 1: USING WISEFROG TO PREDICT LBS FROM PDB ENTRY 1N07

As said before, the user should always follow the standard command line:

```
wisefrog -f ./1n07.pdb -c
```

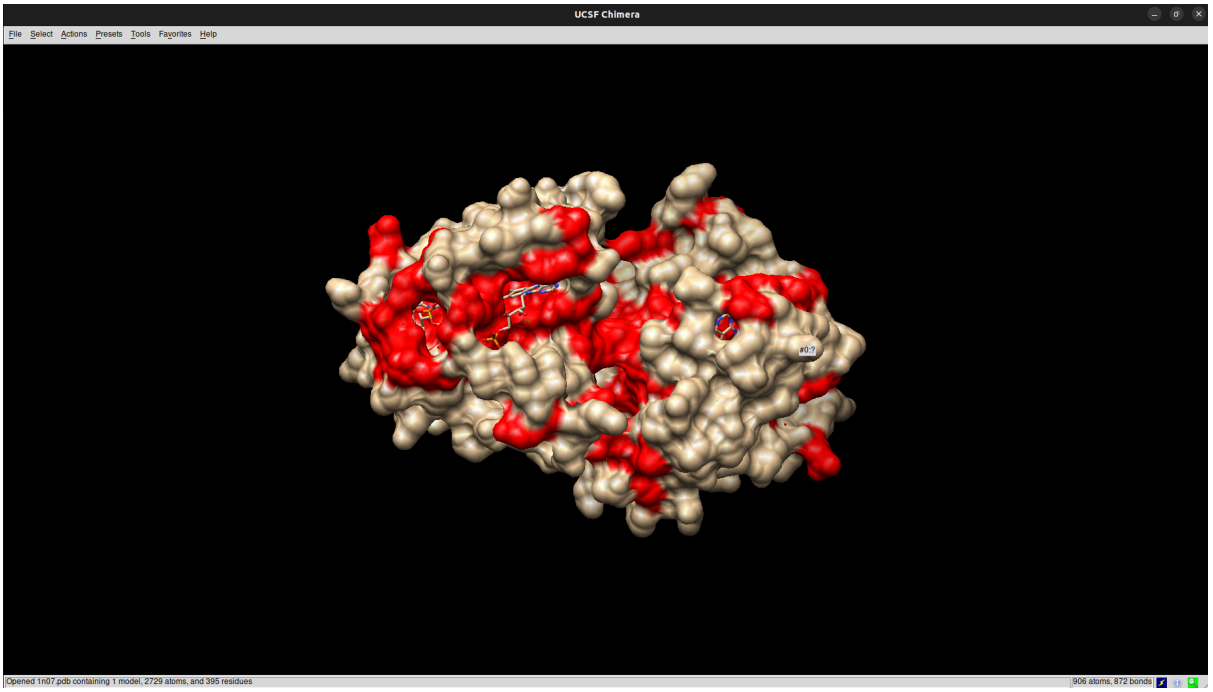
This very simple command categorizes which residue belongs to the LBS of the proteins, adding the chimera flag the output will look like this:

```
========File: 1n07.pdb========
RESIDUE:

ILE12
GLY14
PRO15
GLN19
SER20
TYR22
ILE24
ARG25
PHE26
HIS32
GLY35
ARG36
SER38
...
You can use this selection in chimera to visualize the binding atoms:

:12@CG2,:14@N,:14@CA,:14@C,:14@O,:15@N,:15@CA,:19@CG,:19@OE1,:19@NE2,:20
@C,:20@O,:20@OG,:22@CD1,:22@CE1,:22@CZ,:22@OH,:24@CG1,:25@O,:26@CA,:26@C
E2,:32@NE2,:35@N,:36@CB,:36@CD,:36@NE,:36@CZ,:36@NH1,:36@NH2,:38@CB,:38@
```

OG,:39@CB,:39@CG,:39@CE,:42@O,:44@CA,:44@C,:44@O,:44@CB,:44@CG,:45@N,:45@CA,:45@C,:45@O,:45@CB,:45@OG1,:45@CG2,:46@N,:46@CA,:46@C,:46@CB,:47@CG,:47@OD1,:47@ND2,:52@O,:56@O,:56@CG,:56@CD1,:56@CD2,:59@CA,:59@C,:59@O
...



EXAMPLE 2: USING WISEFROG TO PREDICT LBS FROM bHLH FAMILY

In this example, we'll study many of the bHLH proteins, you'll find the group of PDBs in the following link.

```
wisefrog -f bHLH_family/ -o bHLH_output.out
```

There won't be an output shown in the terminal as it is redirected to the desired output file, thus, the file should look like this:

```
========File: bHLH_family/4rqw.pdb========

RESIDUE:

LEU52

GLN53

LEU59

GLY64

TRP67

…

========File: bHLH_family/7fdn.pdb========

RESIDUE:

LEU13

ARG22

SER27

TYR28

GLY29

…

========File: bHLH_family/1a0a.pdb========

RESIDUE:

MET0

ARG2

HIS5

GLU9

…
```

EXAMPLE 3: USING WISEFROG FOR A SPECIFIC PREDICTION OF LBS

Sometimes the user may require deeper knowledge and information about which atoms are participating in the binding. Since wisefrog is based on the proximity of atoms to the ligand of the training dataset structures, it can provide atom level prediction:

```
wisefrog -f web/1n07 -a
```

```
Downloading from https://files.rcsb.org/download/1n07.pdb
========File: 1n07.pdb========

ATOM        RESIDUE
:12@CG2     ILE12
:14@N       GLY14
:14@CA      GLY14
:14@C       GLY14
:14@O       GLY14
:15@N       PRO15
:15@CA      PRO15
:19@CG      GLN19
:19@OE1     GLN19
:19@NE2     GLN19
:20@C       SER20
:20@O       SER20
:20@OG      SER20
:22@CD1     TYR22
:22@CE1     TYR22
:22@CZ      TYR22
...
You can use this selection in chimera to visualize the binding atoms:

:12@CG2,:14@N,:14@CA,:14@C,:14@O,:15@N,:15@CA,:19@CG,:19@OE1,:19@NE2,:20
@C,:20@O,:20@OG,:22@CD1,:22@CE1,:22@CZ,:22@OH,:24@CG1,:25@O,:26@CA,:26@C
E2,:32@NE2,:35@N,:36@CB,:36@CD,:36@NE,:36@CZ,:36@NH1,:36@NH2,:38@CB,:38@
OG,:39@CB,:39@CG,:39@CE,:42@O,:44@CA,:44@C,:44@O,:44@CB,:44@CG,:45@N,:45
@CA,:45@C,:45@O,:45@CB,:45@OG1,:45@CG2,:46@N,:46@CA,:46@C,:46@CB,:47@CG,
:47@OD1,:47@ND2,:52@O,:56@O,:56@CG,:56@CD1,:56@CD2,:59@CA,:59@C
...
```

Take into account that the atom nomenclature is the one used in chimera : [number of the residue] @ [atom name]. Which is also handy in order to get a better visualization experience.

ERROR HANDLING

One of the errors the user may confront is not having Chimera installed, in that case we hand its documentation in order to get proper information: USCF Chimera. If this is the case, and the user uses the flag `-c` or `--chimera`, he or she will be presented with the message "`Chimera is not installed. Please install Chimera to visualize the binding atoms.`".

Moreover when analyzing very complex structures the model may suffer, which may lead to predicting errors, the team may debug in future versions.