

Feb 23, 2023

## High Throughput Ligand Interaction Profiler

DOI

[dx.doi.org/10.17504/protocols.io.4r3l27njgg1y/v1](https://dx.doi.org/10.17504/protocols.io.4r3l27njgg1y/v1)

Akshay Uttarkar<sup>1</sup>, Shreya Girish<sup>1</sup>, Vidya Niranjana<sup>1</sup>

<sup>1</sup>R V College of Engineering



Vidya Niranjana

R V College of Engineering

### Create & collaborate more with a free account

Edit and publish protocols, collaborate in communities, share insights through comments, and track progress with run records.

Create free account

OPEN  ACCESS



DOI: <https://dx.doi.org/10.17504/protocols.io.4r3l27njgg1y/v1>

**Protocol Citation:** Akshay Uttarkar, Shreya Girish, Vidya Niranjana 2023. High Throughput Ligand Interaction Profiler .  
**protocols.io** <https://dx.doi.org/10.17504/protocols.io.4r3l27njgg1y/v1>

**License:** This is an open access protocol distributed under the terms of the **Creative Commons Attribution License**, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited

**Protocol status:** Working

We use this protocol and it's working

**Created:** January 15, 2023

**Last Modified:** February 23, 2023

**Protocol Integer ID:** 75343

**Keywords:** high throughput ligand interaction profiler, ligand interaction study, large chemical libraries for potential drug candidate, ligand, protein interaction, key residues for ligand, large chemical library, drug discovery, interacting residue, potential drug candidate, protein, google colab, binding site

## Abstract

High Throughput Ligand Interaction Profiler (HT-LIP) is a web-based tool that runs on Google Colab, which allows users to predict ligand-protein interactions. HT-LIP can be used to screen large chemical libraries for potential drug candidates and can also provide insights into potential ligand-protein interactions and identifies the interacting residues. HT-LIP can be useful in identifying potential binding sites and key residues for ligand-protein interactions, which can aid in drug discovery and protein-ligand interaction studies.

## Troubleshooting



## Prepare the Google Drive

- 1 In your Google Drive home directory, create a new folder called "data". Inside this folder, add 3 files.
  - (a) protein: in PDB format
  - (b) ligand: in SDF format
  - (c) reference ligand: in PDB format

For example, the data folder created here, have files with the following names :

- (a) protein: 5Y15.pdb
- (b) ligand: compounds\_for\_ml.sdf
- (c) reference: reference\_ligand.pdb

## Connect to Google Drive

- 2 **Click on the URL provided below to get started.**

URL: [Google Colab Notebook](#)

Now, you will be directed towards the Google Colab Notebook. Towards the LHS side of the notebook, from the "files" section, click on the "mount drive" option. Then, a cell will be created, run the cell. Then click on "Run Anyway" and select "Connect to Google Drive" and give access. Now from the "drive" option, check whether the "data" folder created in the above step is seen in "MyDrive".

## Run the cells

- 3 Below are the seven steps that need to be followed in order to get the Protein-Ligands Interactions.

## ▸ Step 1: Install RDKit software

 [Show code](#)

## ▸ Step 2: Install dependencies

 [Show code](#)

## ▸ Step 3: Load protein(in PDB format)

 `Folder_name: "/content/drive/MyDrive/data/5Y15.pdb"`[Show code](#)

## ▸ Step 4: Print smiles format

 `Folder_name: "/content/drive/MyDrive/data/compounds_for_ml.sdf"``sdf: "Chem.SDMolSupplier( '/content/drive/MyDrive/data/compounds_for_ml.sdf' )"`

## ▸ Step 5: Load ligands(in SDF format)

 `Folder_name: "/content/drive/MyDrive/data/compounds_for_ml.sdf"`[Show code](#)

## ▸ Step 6: Load reference(in PDB format)

 `Folder_name: "/content/drive/MyDrive/data/reference_ligand.pdb"`[Show code](#)

## ▸ Step 7: Download excel format for the interactions

 [Show code](#)**Step 1: Install RDKit software**

RDKit is an open source Chemoinformatics & Machine Learning Software. It provides tools for different kinds of similarity search. To install RDKit, click on the run cell option beside the show code(in blue).

## ► Step 1: Install RDKit software

[Show code](#)

In a few seconds RDKit will be installed.

```
+ Code + Text RAM 80 MB Disk 100 MB Editing

Step 1: Install RDKit software

@title Step 1: Install RDKit software

!pip install rdkit-pypi prolif

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting rdkit-pypi
  Downloading rdkit_pypi-2022.9.3-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (29.3 MB)
    29.3/29.3 MB 37.2 MB/s eta 0:00:00
Collecting prolif
  Downloading prolif-1.1.0-py3-none-any.whl (5.1 MB)
    5.1/5.1 MB 43.0 MB/s eta 0:00:00
Requirement already satisfied: Pillow in /usr/local/lib/python3.8/dist-packages (from rdkit-pypi) (7.1.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.8/dist-packages (from rdkit-pypi) (1.21.6)
Requirement already satisfied: scipy>=1.3.0 in /usr/local/lib/python3.8/dist-packages (from prolif) (1.7.3)
Collecting mdanalysis>=2.2.0
  Downloading MDAnalysis-2.4.2-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (8.1 MB)
    8.1/8.1 MB 47.6 MB/s eta 0:00:00
Requirement already satisfied: pandas>=1.0.0 in /usr/local/lib/python3.8/dist-packages (from prolif) (1.3.5)
Requirement already satisfied: tqdm in /usr/local/lib/python3.8/dist-packages (from prolif) (4.64.1)
Collecting biopython>=1.80
  Downloading biopython-1.80-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (3.1 MB)
    3.1/3.1 MB 80.5 MB/s eta 0:00:00
Requirement already satisfied: matplotlib>=1.5.1 in /usr/local/lib/python3.8/dist-packages (from mdanalysis>=2.2.0->prolif) (3.2.2)
Collecting gsd>=1.9.3
  Downloading gsd-2.7.0-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (406 kB)
```

## **Step 2: Install dependencies**

Click on the run cell option beside the show code(in blue) to install the dependencies(libraries). The required libraries will be installed in a few seconds.



## **Step 3: Load protein(in PDB format)**

The protein used here is 5Y15. To upload your protein of interest, change "5Y15" to that protein name in the "folder\_name" section. Now run the cell.

### ► Step 3: Load protein(in PDB format)



Folder\_name: `"/content/drive/MyDrive/data/5Y15.pdb"`

[Show code](#)

### **Step 4: Print smiles format**

Change "compounds\_for\_ml" to your ligand name in the "folder\_name" and "sdf" section.  
Now run the cell.

Here ligands are converted from SDF to SMILES format.

#### Step 4: Print smiles format



Folder\_name: `"/content/drive/MyDrive/data/compounds_for_ml.sdf"`

sdf: `Chem.SDMolSupplier('/content/drive/MyDrive/data/compounds_for_ml.sdf')`

[Show code](#)

The smiles format will be printed.

#### Step 4: Print smiles format



```
title Step 4: Print smiles format
on rdkit import Chem

folder_name = "/content/drive/MyDrive/data/compounds_for_ml.sdf" #@para
f = Chem.SDMolSupplier('/content/drive/MyDrive/data/compounds_for_ml.sdf')
with open('smiles.smil', 'w') as f:
    for mol in sdf:
        smi = Chem.MolToSmiles(mol)
        print(smi)
```

```
O=C([O-])C(=O)O
O=S(=O)([O-])[N-]S(=O)(=O)[O-]
O=S(=O)([O-])OS(=O)(=O)[O-]
O=C([O-])[C@H]1C(=O)OCCO1
O=C([O-])C(=O)C[C@H](O)C(=O)[O-]
O=C([O-])[C@H](O)C(O)[C@H](O)C(=O)[O-]
[NH3+][C@H](C[C@H](C(=O)[O-])C(=O)O)C(=O)[O-]
O=C(O)[C@H]1C[C@H]1C(=O)[O-]C(=O)O
O=C([O-])CS(=O)CC(=O)[O-]
O=C(O)CCC[C@H](O)C(=O)[O-]
C[C@H](O)C(=O)[O-]1C[C@H](O)C(=O)O1
```

Folder\_name: `"/content/drive/MyDrive/data/compounds_for_ml.sdf"`

sdf: `Chem.SDMolSupplier('/content/drive/MyDrive/data/compounds_for_ml.sdf')`



### Step 5: Load ligands(in SDF format)

Change "compounds\_for\_ml" to your ligand name in the "folder\_name" section(same as step 4). Run the cell.

#### Step 5: Load ligands(in SDF format)



Folder\_name: `"/content/drive/MyDrive/data/compounds_for_ml.sdf"`

Show code

Following output will be obtained.



The screenshot shows a Jupyter Notebook cell with a code editor at the top containing four SMILES strings. Below the code editor is a table with 8 rows and 8 columns. The first two columns are 'Ligand' and 'UNL1'. The next six columns are protein names: 'GLN55.A', 'ASP72.A', 'LYS104.A', 'ASN105.A', 'ARG107.A', and 'ARG109.A'. The last column is 'smiles'. The table is titled 'Interaction' and 'Frame'.

Ligand	UNL1	GLN55.A	ASP72.A	LYS104.A	ASN105.A	ARG107.A	ARG109.A	smiles
0	False	False	False	True	False	True		O=C([O-])C(=O)O
1	True	False	True	False	False	False		O=S(=O)([O-])[N-]S(=O)(=O)[O-]
2	True	False	False	False	False	False		O=S(=O)([O-])OS(=O)(=O)[O-]
3	True	False	True	False	False	False		O=C([O-])C([O-])C(=O)OCCO1
4	False	True	False	True	False	True		O=C([O-])C(=O)C([O-])C(=O)O
5	False	False	True	False	False	False		O=C([O-])C([O-])C(=O)O
6	False	False	True	False	False	False		[NH3+][C@@H](C)[C@@H](C(=O)O)C(=O)O
7	False	False	True	False	False	False		O=C([O-])C([O-])C(=O)O

### Step 6: Load reference(in PDB format)

Change "reference\_ligand" to your ligand name in the "folder\_name" section. Now run the cell.

## ► Step 6: Load reference(in PDB format)



Folder\_name: `"/content/drive/MyDrive/data/reference_ligand.pdb"`

[Show code](#)

Following output will be obtained.

Step 6: Load reference(in PDB format) ↑ ↓ ↻ 📄 ⚙️

Folder\_name: `"/content/drive/MyDrive/data/reference_ligand.pdb"`

[Show code](#)

100% 1/1 [00:00<00:00, 9.60it/s]

protein	ARG107.A	ARG109.A	ASN105.A	ASP72.A	GLN55.A	LYS104.A
interaction	VdwContact	VdwContact	VdwContact	VdwContact	VdwContact	VdwContact

Frame	ref	False	False	True	True	True	True	False
0	<chem>O=C([O-])C(=O)O</chem>	False	True	True	False	False	False	False
1	<chem>O=S(=O)([O-])[N-]S(=O)(=O)[O-]</chem>	False	False	False	False	True	True	True
2	<chem>O=S(=O)([O-])OS(=O)(=O)[O-]</chem>	False	False	False	False	True	False	False
3	<chem>O=C([O-])[C@]1(C(=O)O)CCC1</chem>	False	False	False	False	True	True	True
4	<chem>O=C([O-])C(=O)C[C@H](O)C(=O)[O-]</chem>	False	True	True	True	False	False	False

## ***Step 7: Download excel format for the interactions***

Download the excel file by running the cell. The downloaded excel file will be saved in the file section(LHS of colab notebook).





## ► Step 7: Download excel format for the interactions

[Show code](#)

### Citation

Bouysset C, Fiorucci S (2021)

. ProLIF: a library to encode molecular interactions as fingerprints..  
Journal of cheminformatics.

<https://doi.org/10.1186/s13321-021-00548-6>

LINK

. J Cheminform**13**, 72 (2021). <https://doi.org/10.1186/s13321-021-00548-6>