PYCON Indonesia 2020

Machine Learning Based Drug Discovery & Repurposing

By: Aries Fitriawan S.Komp M.Kom

PyCon Indonesia 2020, Nov 14th – 15th

ARIES FITaRIAWAN

Data Scientist, PT XL Axiata tbk. (??????????????)

Adjunct Researcher, IPB - IMERI UI

Education

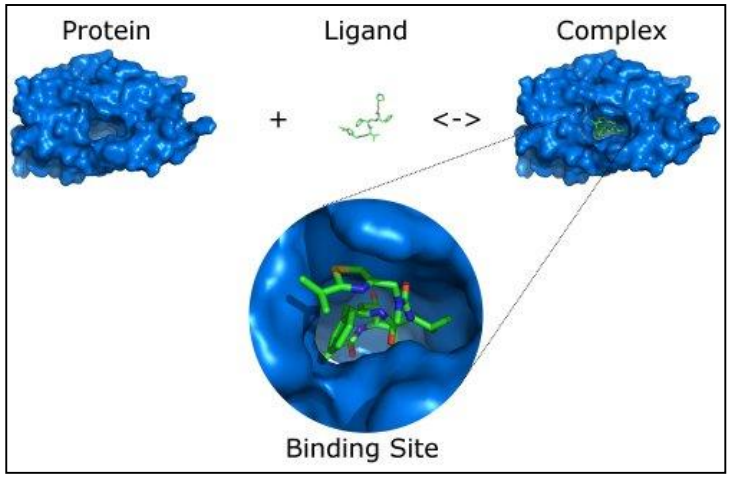
2014 - 2016 **Universitas Indonesia**
Master Degree of Computer Science, Bioinformatics Specialization.

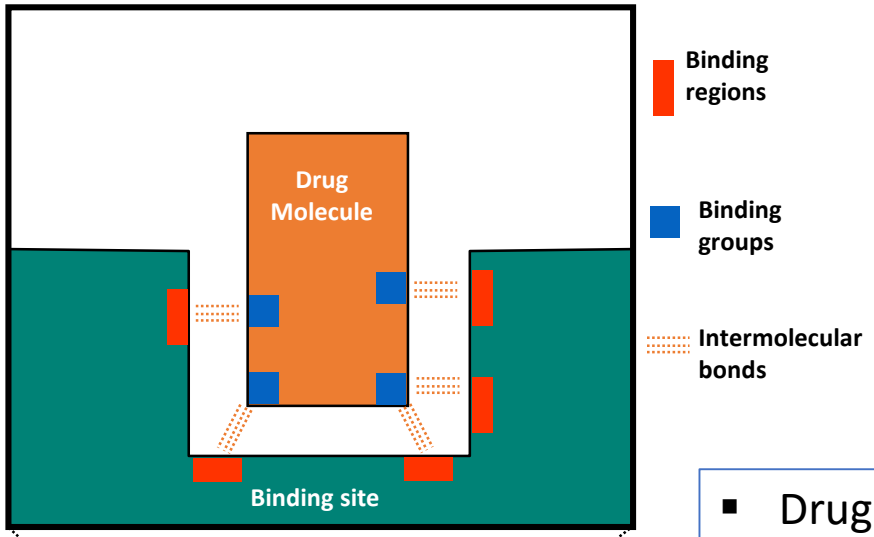
2009 - 2013 **Institut Pertanian Bogor**
Bachelor Degree of Computer Science.

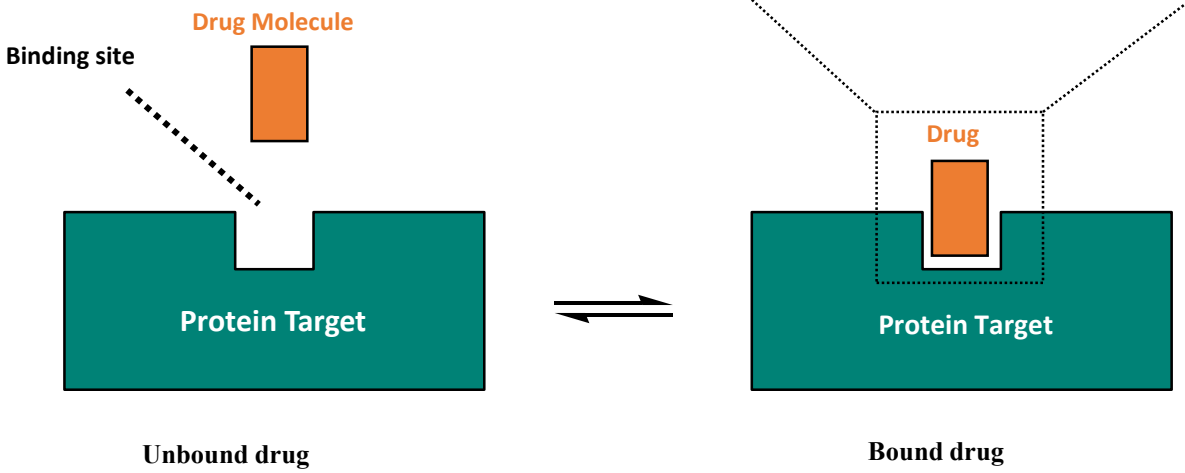
Publication

- ✓ Virtual Screening on Indonesian Herbal Compounds as COVID-19 Supportive Therapy: Machine Learning and Pharmacophore Modeling Approaches (pre-print, 2020)
- ✓ Deep Belief Networks Using Hybrid Fingerprint Feature for Virtual Screening of Drug Design (2016)
- ✓ Multimodal Deep Boltzmann Machines For Feature Selection on Gene Expression Data (2016)
- ✓ Multi-Label Classification Using Deep Belief Networks for Virtual Screening of Multi Target Drug (2016)
- ✓ Cancer Subtype Identification Using Deep Learning Approach (2016)
- ✓ Identification of Gene Expression Linked to Malignancy of Human Colorectal Carcinoma using Restricted Boltzmann Machines (2016)
- ✓ Deep Belief Networks for Ligand-Based Virtual Screening of Drug Design (2016)
- ✓ Support Vector Machine OVA-RFE Approach for Finding the Significant Plants of Jamu (2016)
- ✓ A classification system for Jamu efficacy based on formula using Support Vector Machine (2013)

Drug Molecule Mechanism

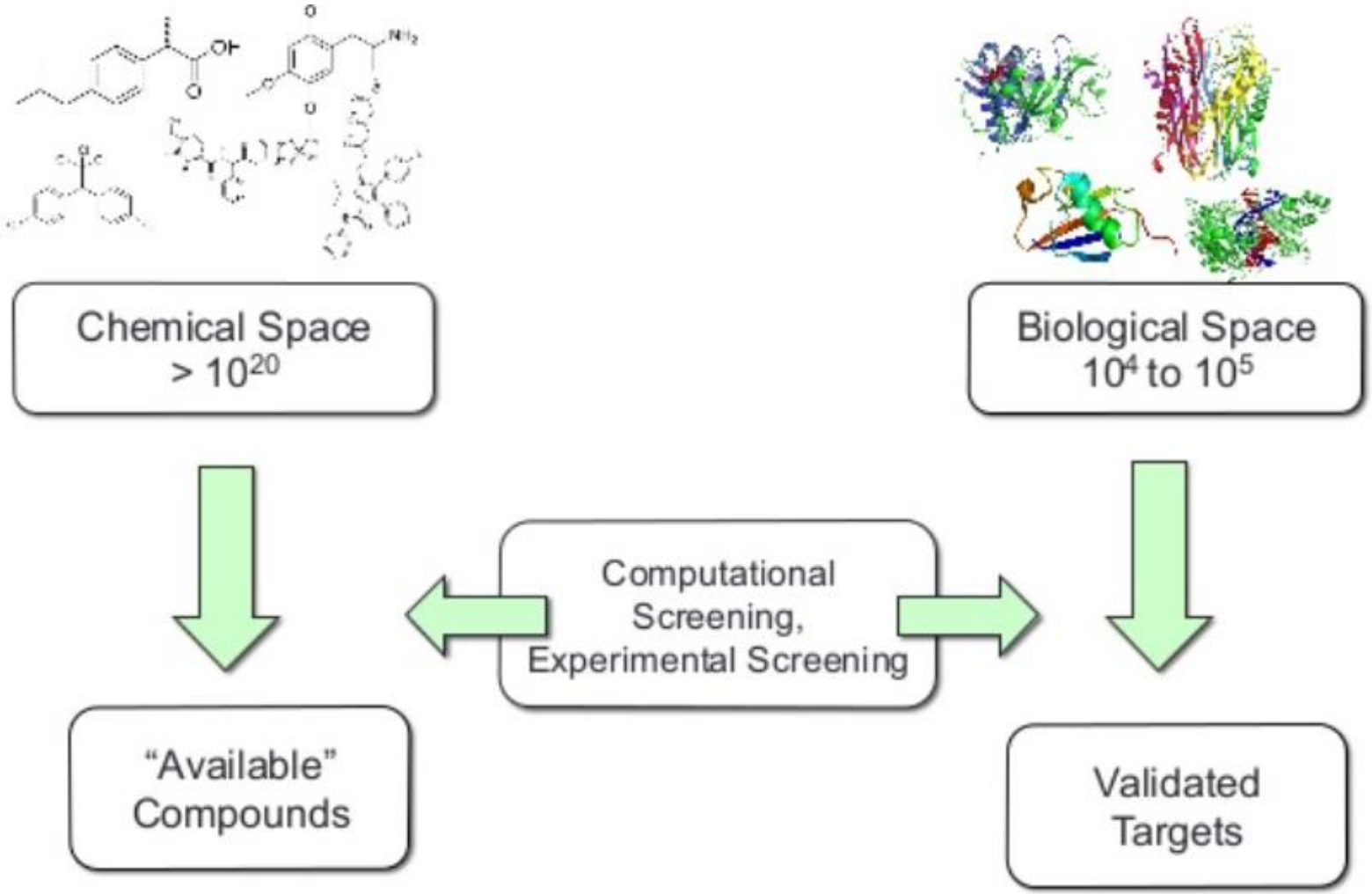






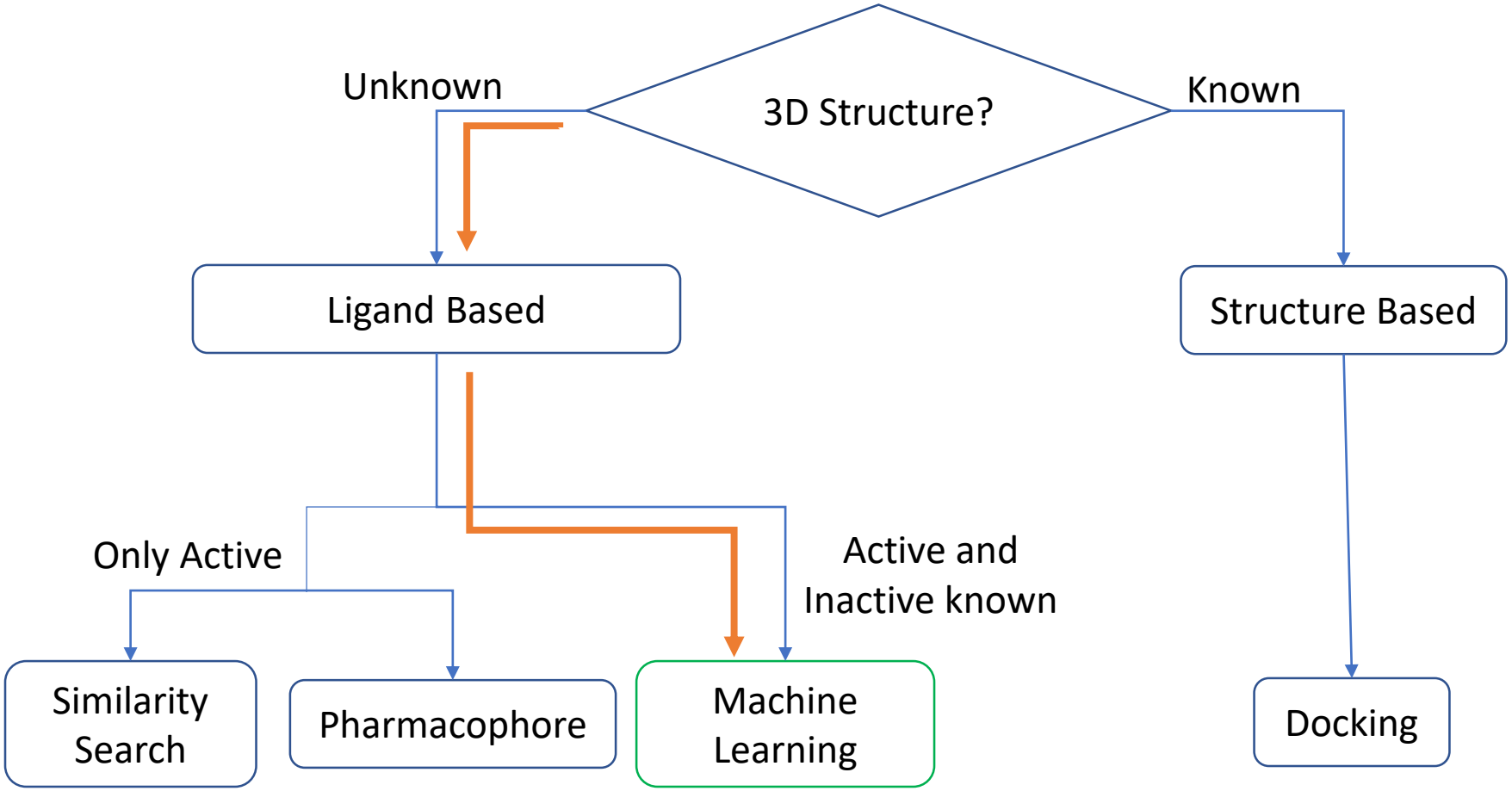
- Drug targets are large protein molecules - macromolecules
- Drugs are generally much smaller than their targets
- Drugs interact with their targets by binding-to-binding sites
- **With Machine Learning, we can predict whether drug molecule can bind with the target or not**
- **Active drug compounds mean there is interaction between drug and protein**

The Challenge



Blum, L.C. & Reymond, J.-louis . *J. Am. Chem. Soc*. 131, 8732-8733(2009).

What is Virtual Screening?



Depending upon structural and Bioactvity data available :

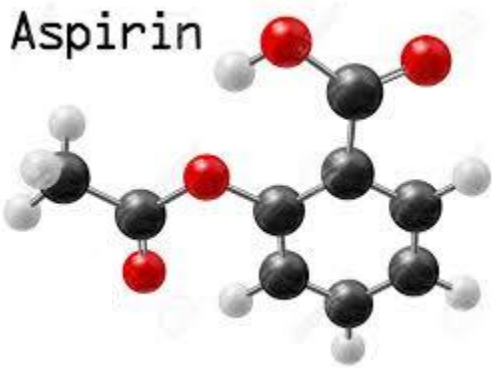
- ✓ One or more actives molecule known perform **similarity searching.**
- ✓ Several active known try to identify a common 3D **pharmacophore** and then do 3D database search.
- ✓ Reasonable number of active and inactive known train a **machine learning** model (Big Data).
- ✓ 3D structure of protein known use protein ligand **docking**.

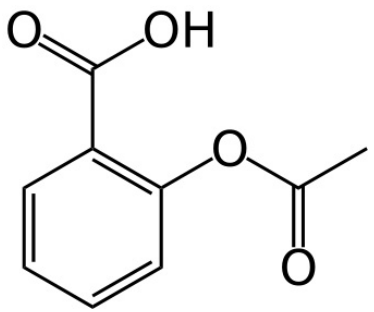
Computer Representation of Compound

STRING

- Common names: aspirin
- IUPAC name: 2-acetoxybenzoic acid
- Formula: $C_9H_8O_4$
- CAS number: 50-78-2
- SMILES (simplified molecular-input line-entry system) string:
  O=C(Oc1ccccc1C(=O)O)C
- File format: ChemDraw file, MOL file, etc.

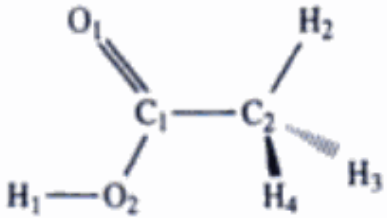
IMAGE



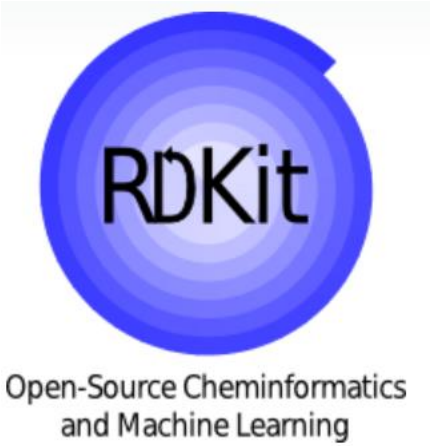


GRAPH

The structure of a molecule can be represented by a **graph**
Graph = collection of nodes and edges, nodes and edges have properties (atomic number, bond order)



What Is RDKIT ?



http://www.rdkit.org

Open-source toolkit for cheminformatics

- Business-friendly BSD license
- Core data structures and algorithms in C++
- Python 3.x wrappers generated using Boost.Python
- 2D and 3D molecular operations
- Descriptor generation for machine learning
- Molecular database cartridge for PostgreSQL
- Cheminformatics nodes for KNIME (distributed from the KNIME community site: https://www.knime.com/rdkit)

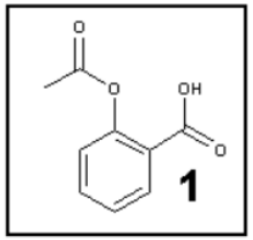
Installation

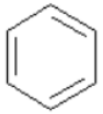
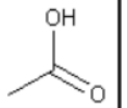
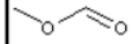
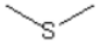
Install RDKit by using these commands

```
conda install libboost=1.65.1
conda install boost=1.65.1
conda install boost-cpp=1.65.1
conda install -c rdkit rdkit
```

Molecular Fingerprint Representation for Compound

- ✓ **Molecular fingerprints** are a way of encoding the structure of a **molecule**.
- ✓ The most common type of **fingerprint** is a series of binary digits (bits) that represent the presence or absence of particular substructures in the **molecule**.
- ✓ There are 3 different type of fingerprint: *substructure keys-based, path-based, dan circular fingerprint*
- ✓ Most common substructure key-based fingerprints are **PubChem (881 fingerprints)** and Klekota-Roth fingeprints (4860 fingerprints)

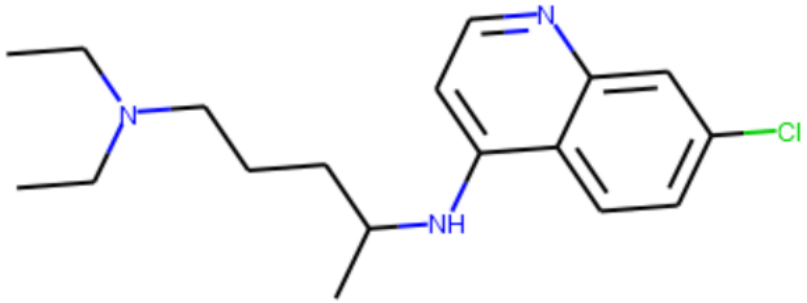


1	1	0	1	1	0	1	0
							

Inside RDKIT Library

- Read and Draw the molecules

```
#Read SMILES text data as a "mol" and draw the 2d molecules structure
mol = Chem.MolFromSmiles("CCN(CC)CCCC(C)Nc1ccnc2cc(Cl)ccc12")
mol
```



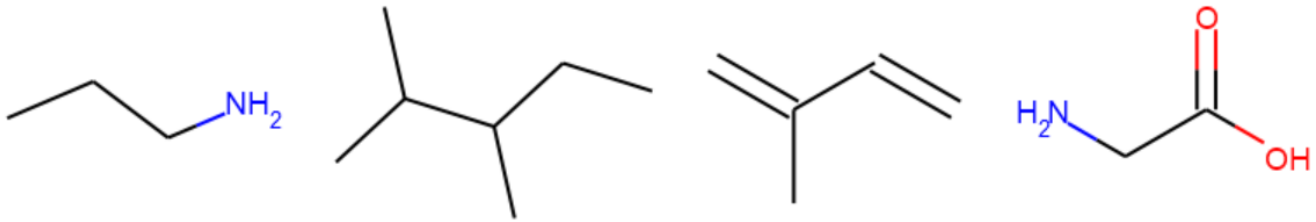

```
print(mol)
```

```
<rdkit.Chem.rdchem.Mol object at 0x0000023A2DD1FD50>
```

```
#Change back mol into SMILES
smiles =Chem.MolToSmiles(mol)
smiles
```

```
'CCN(CC)CCCC(C)Nc1ccnc2cc(Cl)ccc12'
```

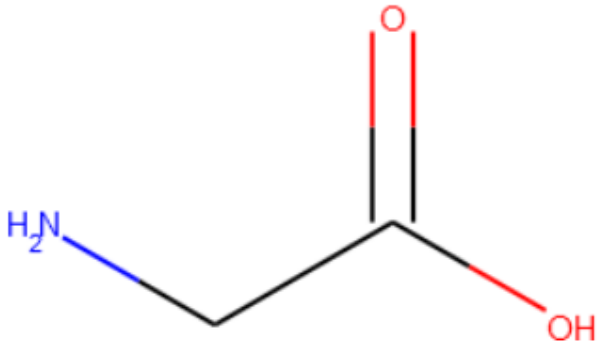
```
#Draw group of molecule
smiles_list = ["CCCN", "C(C)(C)C(C)C(C)", "C=C(C)C=C", "C(C(=O)O)N"]

mol_list = []

for x in smiles_list:
    mol = Chem.MolFromSmiles(x)
    mol_list.append(mol)

img = Draw.MolsToGridImage(mol_list, molsPerRow=4)
img
```



Inside RDKIT Library: Finding Pattern

```
#Finding Pattern of Substructure
pattern = Chem.MolFromSmiles("CN")
pattern
```


```
mol = Chem.MolFromSmiles("C(C(=O)O)N")
mol
```

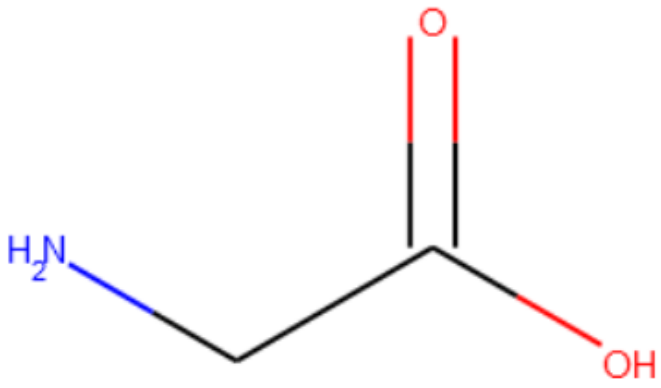



```
print(mol.HasSubstructMatch(pattern))
```

```
True
```

Inside RDKIT: Morgan Fingerprint Extraction

```
mol = Chem.MolFromSmiles("C(C(=O)O)N")
mol
```



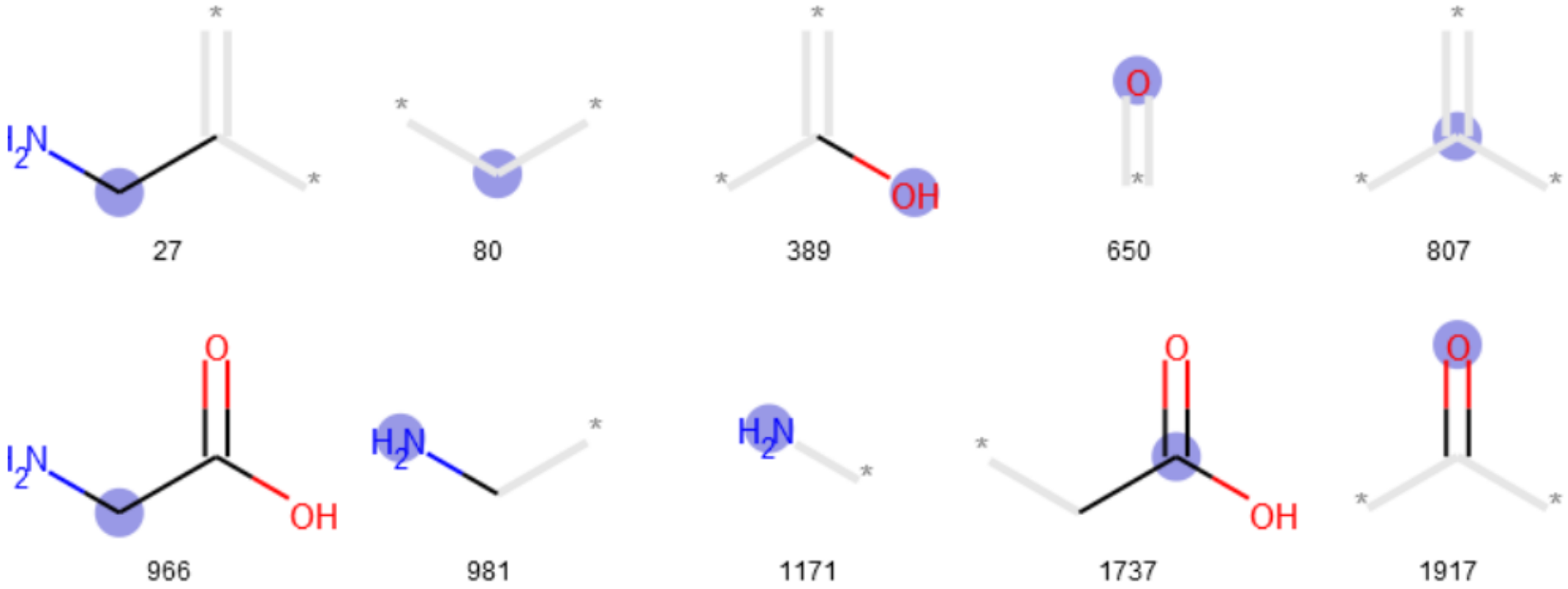

```
#draw A fingerprints
bi = {}
mol = Chem.MolFromSmiles("C(C(=O)O)N")
fp = AllChem.GetMorganFingerprintAsBitVect(mol, 2, nBits = 2048, bitInfo = bi)

#create empty list
fp_arr = np.zeros((1),)

#input the fingerprint index into list
DataStructs.ConvertToNumpyArray(fp,fp_arr)
np.nonzero(fp_arr)
```

```
(array([  27,   80,  389,  650,  807,  966,  981, 1171, 1737, 1917],
       dtype=int64),)
```

Inside RDKIT: Draw Morgan Fingerprints Structure

```
fp = AllChem.GetMorganFingerprintAsBitVect(mol, 2, nBits = 2048, bitInfo = bi)

prints = [(mol,x,bi) for x in fp.GetOnBits()]

#Draw FP list location
Draw.DrawMorganBits(prints, molsPerRow=5, legends = [str(x) for x in fp.GetOnBits()])
```

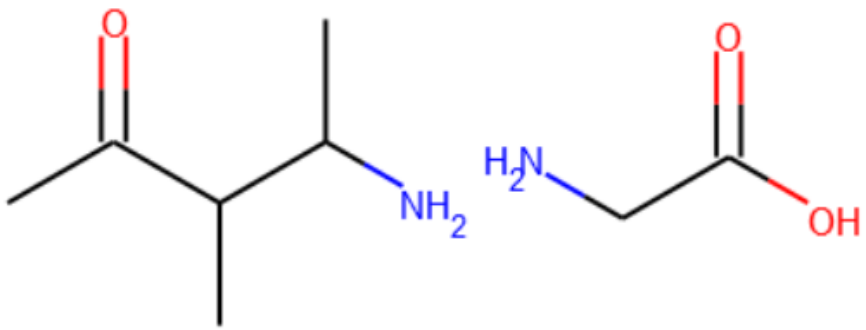
```
(array([  27,   80,  389,  650,  807,  966,  981, 1171, 1737, 1917],
       dtype=int64),)
```



Inside RDKIT: Tanimoto Index for Molecule Similarity

1. Read Data

```
smiles_list = ["NC(C)C(C)C(C)(=O)", "C(C(=O)O)N"]
mol_list = []

for x in smiles_list:
    mol = Chem.MolFromSmiles(x)
    mol_list.append(mol)

img = Draw.MolsToGridImage(mol_list, molsPerRow=4)
img
```



2. Extract Fingerprints

```
#Extract Fingerprints
fp1 = AllChem.GetMorganFingerprintAsBitVect(mol_list[0], 2, nBits = 2048, bitInfo = bi)
fp2 = AllChem.GetMorganFingerprintAsBitVect(mol_list[1], 2, nBits = 2048, bitInfo = bi)
```

```
print("1.", list(fp1.GetOnBits()))
print("2.", list(fp2.GetOnBits()))
```

```
1. [1, 283, 299, 403, 507, 633, 650, 786, 807, 1017, 1057, 1171, 1497, 1832, 1917]
2. [27, 80, 389, 650, 807, 966, 981, 1171, 1737, 1917]
```

3. Calculate Tanimoto Index ($\frac{|A \cap B|}{|A \cup B|}$)

```
#calculate Tanimoto Similarity Index (AnB)/(AuB)
print(DataStructs.TanimotoSimilarity(fp1,fp2)*100,"%")
```

```
19.047619047619047 %
```

PyFingerprint for massive chemical fingerprint types

HC.Ji
hcji

I' m Hongchao Ji. A postdoctoral researcher working on MS-based data analysis method development for proteomics and metabolomics.

Follow ···

https://github.com/hcji/PyFingerprint

There are many types of chemical fingerprint for describing the molecule provided by different tools, such as RDKit, CDK and OpenBabel. This package aims to summarize them all.

Dependencies

- Anaconda for python 3.6
- Java Runtime Environment 8.0
- jpype
- RDKit

Installation

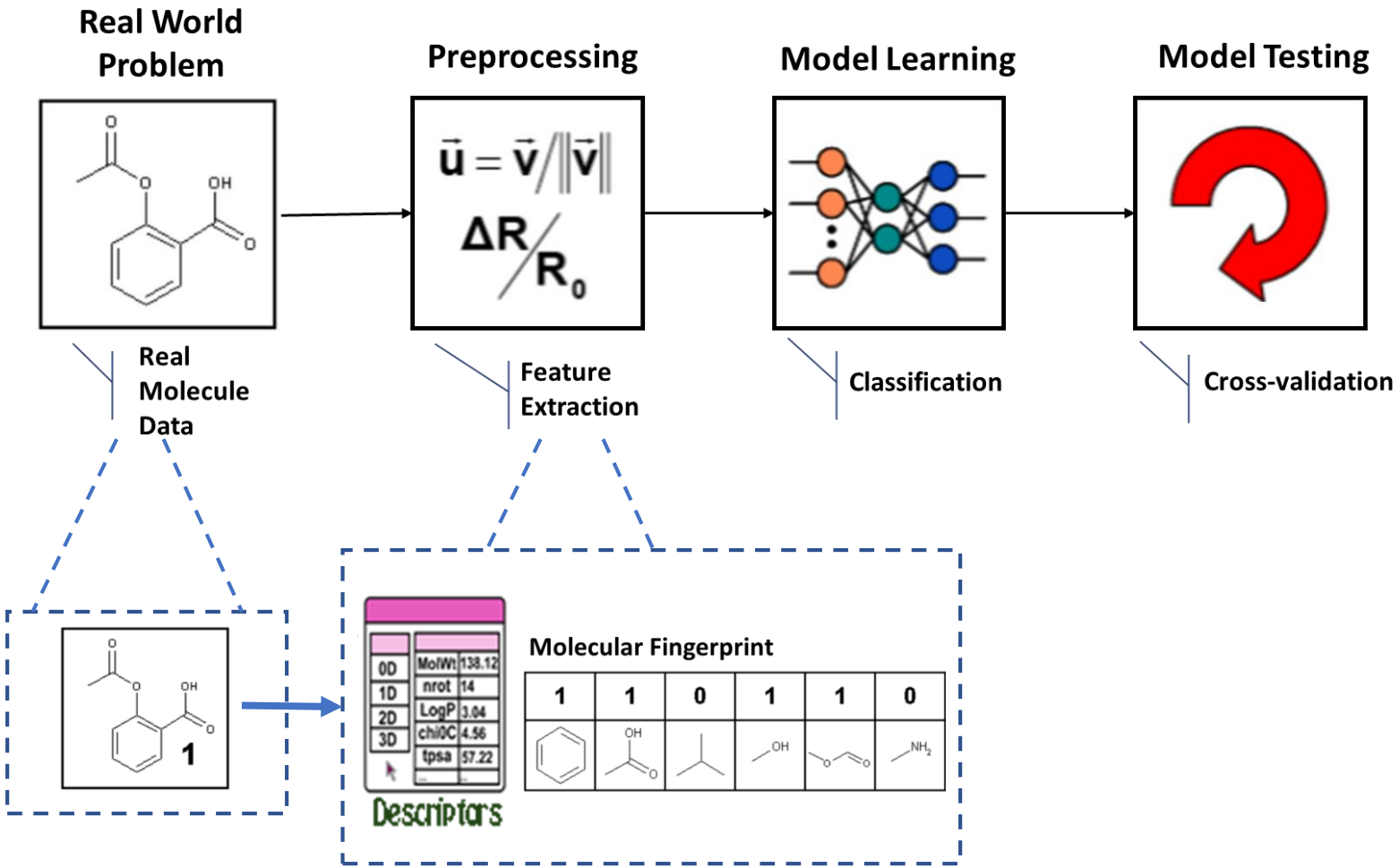
```
pip install git+git://github.com/hcji/PyFingerprint@master
```

Usage

```python
from PyFingerprint.All_Fingerprint import get_fingerprint

fps1 = get_fingerprint('C(C(=O)O)N', fp_type='pubchem')
fps2 = get_fingerprint('C(C(=O)O)N', fp_type='klekota-roth')
print("Pubchem fp:",fps1,"\n")
print("Klekota-Roth fp:",fps2)
```

```
Pubchem fp: [9, 14, 18, 19, 283, 284, 285, 286, 299, 308, 344, 345, 351, 352, 365, 380, 393, 406, 420, 440, 443, 452, 528, 536,
566]

Klekota-Roth fp: [296, 492, 503, 547, 1145, 1147, 1192, 1240, 1405, 2948, 2974, 3024, 3223, 3327, 3454, 3749, 3787, 3881, 3955,
4079, 4282, 4285, 4294, 4330, 4694]
```

Utilize the Extracted Feature in Almost All Python ML Library



Have you seen these News?

Coronavirus and hydroxychloroquine: What do we know?

By Jack Goodman and Christopher Giles
BBC Reality Check

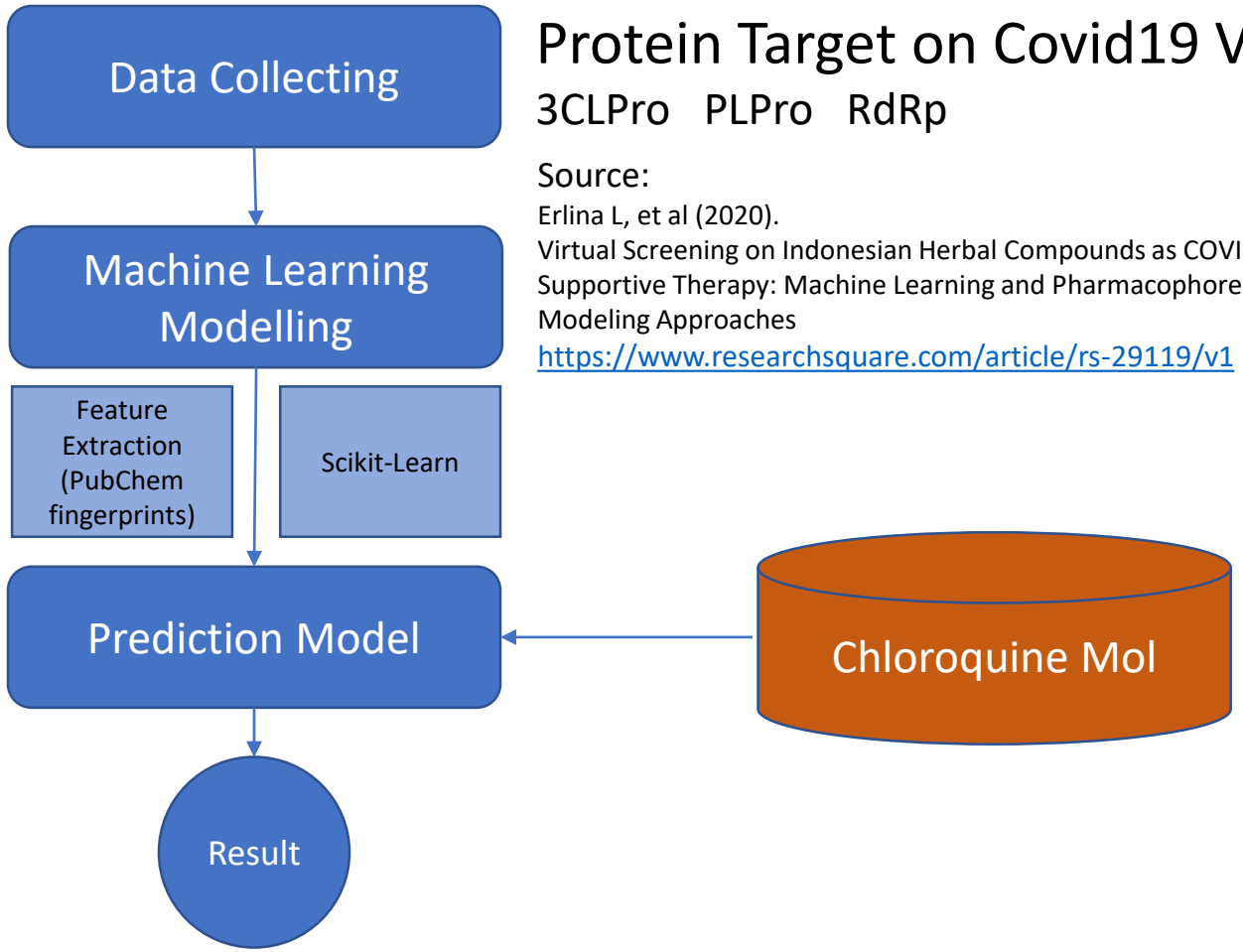
27 July

Reality Check

There are doubts about its effectiveness as a treatment

There's been widespread interest in hydroxychloroquine as both a preventative measure and for treating patients with coronavirus.

Let's Do a simple Exercise in Machine Learning Perspective!



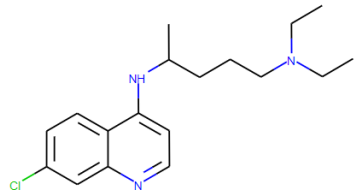
Protein Target on Covid19 Virus

3CLPro PLPro RdRp

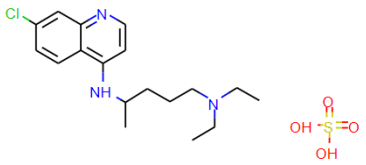
Source:

Erlina L, et al (2020).
Virtual Screening on Indonesian Herbal Compounds as COVID-19 Supportive Therapy: Machine Learning and Pharmacophore Modeling Approaches
https://www.researchsquare.com/article/rs-29119/v1

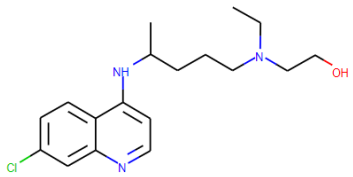
Is It Chloroquine? – Machine Learning Perspective



ID:	CHEMBL76
Name:	CHLOROQUINE
Max Phase:	4 Approved
Molecular Formula:	C18H26ClN3
Molecular Weight:	319.88
ChEMBL Synonyms:	Aralen, CHLOROCHINE, Chloroquine, CHLOROQUINE, NSC-187208
Synonyms From Alternate Forms:	CHLOROQUINE HYDROCHLORIDE, Chloroquine Hydrochloride, Chloroquine Diphosphate, CHLOROQUINE DIPHOSPHATE, Chloroquine Phosphate, CHLOROQUINE PHOSPHATE, NSC-14050, SN-7618, WR-1522, Chloroquine [3H], CHLOROQUINE SULFATE, CHLOROQUINE SULPHATE, NSC-292296



ID:	CHEMBL2095223
Name:	CHLOROQUINE SULPHATE
Max Phase:	4 Approved
Molecular Formula:	C18H28ClN3O4S
Molecular Weight:	417.96
ChEMBL Synonyms:	CHLOROQUINE SULFATE, CHLOROQUINE SULPHATE, NSC-292296
Synonyms From Parent:	Aralen, CHLOROCHINE, Chloroquine, CHLOROQUINE, NSC-187208
Molecule Type:	Small molecule



ID:	CHEMBL1535
Name:	HYDROXYCHLOROQUINE
Max Phase:	4 Approved
Molecular Formula:	C18H26ClN3O
Molecular Weight:	335.88
ChEMBL Synonyms:	Hydroxychloroquine, HYDROXYCHLOROQUINE
Synonyms From Alternate Forms:	Hydroxychloroquine Sulfate, HYDROXYCHLOROQUINE SULFATE, NSC-4375, OXICHLOROCHINE SULFATE, TCMDC-123987
Trade Names From Alternate Forms:	Ercoquin, HYDROXYCHLOROQUINE SULFATE, Oxichlorochine, PLAQUENIL, Quensyl, QUINORIC
Molecule Type:	Small molecule

Probability to Bind with Proteins

	3CLPro	PLPro	RdRp
CHLOROQUINE	0.75%	0.06%	0.03%
CHLOROQUINE SULPHATE	1.24%	0.05%	0.03%
HYDROXYCHLOROQUINE	0.75%	0.06%	0.03%

Disclaimer:

This isn't a conclusion; we need more "**expert touch**" collaboration in the future especially for all **Subject Matter Expert**.

2020 and beyond, a year for collaboration.

Get well soon, World! ☺

You can see all the exploration code here
https://github.com/rietaros/pyconid2020

... يَرْفَعِ ٱللَّهُ ٱلَّذِينَ ءَامَنُوا۟ مِنكُمْ وَٱلَّذِينَ أُوتُوا۟ ٱلْعِلْمَ دَرَجَٰتٍ ۚ وَٱللَّهُ بِمَا تَعْمَلُونَ خَبِيرٌ

... Allah will raise those who have believed among you and **those who were given knowledge**, by degrees. And Allah is Acquainted with what you do.

Al Quran – Al Mujadilah (58 : 11)