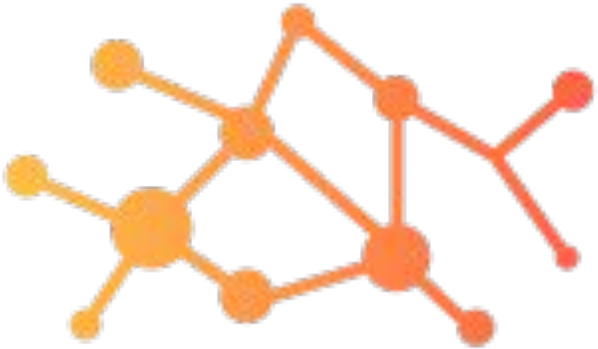


OptiMol

Yunping, Weishi, Vivian, Minh, Guanning

3/17/2020

Background

Motivation

- ❏ Predicting molecular conformation without specialty in quantum mechanic approaches.

Approach

- ❏ Convert 2D drawing into feedable data frame for machine learning model, and return a 3D molecule structure.

Originality

- ❏ Significantly simplified workload compared to DFT
- ❏ Smaller data size for storage

Use cases

Potential users:

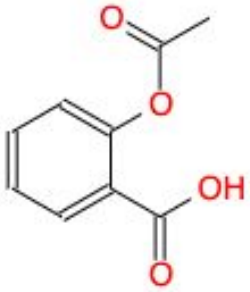
1) Chemical engineers and material scientists who want to obtain the approximate 3D structure of molecule before making a lot of effort synthesizing them.
2) Chemical engineers and material scientists who don't possess a profound quantum mechanics background to use density function theory (DFT).

What is needed?

2D molecule input of X-Y coordinates and atom connections, e.g. Molfile.

What will generate?

Atom coordination in 3D structure including bond length, type and angle, e.g. CIF.



Use SMILES or drawing to create molecule in 2D plane

Atom position

```
13 13  0     0  0            999 V2000
    4.4977   -3.4633    0.0000 C   0  0  0  0  0  0  0       0  0  0
    3.9985   -2.5997    0.0000 C   0  0  0  0  0  0  0       0  0  0
    4.4977   -1.7317    0.0000 O   0  0  0  0  0  0  0       0  0  0
    3.0000   -2.5997    0.0000 O   0  0  0  0  0  0  0       0  0  0
    2.5008   -1.7317    0.0000 C   0  0  0  0  0  0  0       0  0  0
    3.0000   -0.8680    0.0000 C   0  0  0  0  0  0  0       0  0  0
    2.5008    0.0000    0.0000 C   0  0  0  0  0  0  0       0  0  0
    1.4977    0.0000    0.0000 C   0  0  0  0  0  0  0       0  0  0
    0.9985   -0.8680    0.0000 C   0  0  0  0  0  0  0       0  0  0
    1.4977   -1.7317    0.0000 C   0  0  0  0  0  0  0       0  0  0
    0.9985   -2.5997    0.0000 C   0  0  0  0  0  0  0       0  0  0
    0.0000   -2.5997    0.0000 O   0  0  0  0  0  0  0       0  0  0
    1.4977   -3.4633    0.0000 O   0  0  0  0  0  0  0       0  0  0
  1  2  1  0     0  0
  2  3  2  0     0  0
  2  4  1  0     0  0
  4  5  1  0     0  0
  5  6  1  0     0  0
  5 10  2  0     0  0
  6  7  2  0     0  0
  7  8  1  0     0  0
  8  9  2  0     0  0
  9 10  1  0     0  0
 10 11  1  0     0  0
 11 12  1  0     0  0
 11 13  2  0     0  0
```

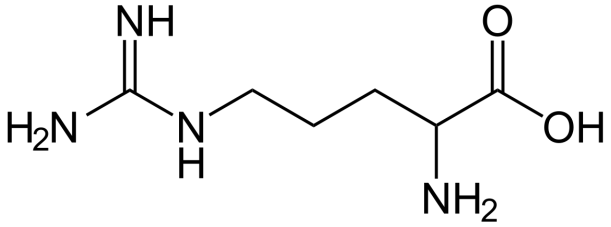
Predict 3D molecule structure

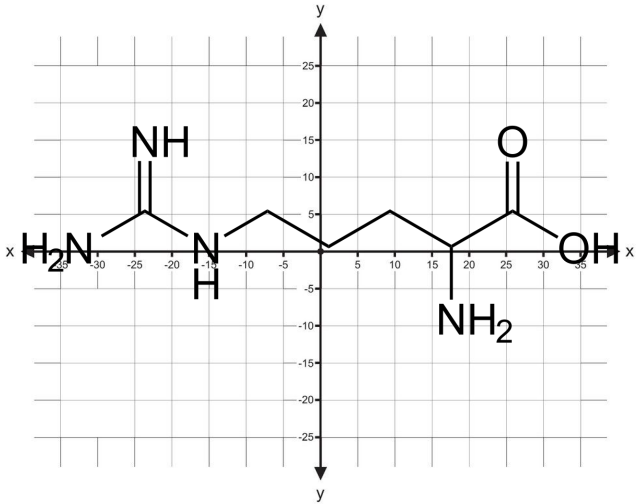
Generate 2D molecule information data frame

Atom connection & bond type

	3d_x	3d_y	3d_z	atom	periodic_#	connect_to	bond_1	bond_2	bond_3
0	1.4472	-1.7071	-0.0590	O	8	[5, -1, -1, -1]	1	0	0
1	2.7285	0.6331	-0.0618	O	8	[6, -1, -1, -1]	1	0	0
2	-2.4979	-1.9721	0.4508	O	8	[10, -1, -1, -1]	1	0	0
3	-0.9625	-2.9028	-0.5393	O	8	[10, 10, -1, -1]	0	1	0
4	-0.6498	-0.6762	-0.0566	C	6	[5, 5, 7, 10]	2	1	0
5	0.7735	-0.6395	-0.0670	C	6	[0, 4, 4, 6]	2	1	0
6	1.4691	0.5934	-0.0639	C	6	[1, 5, 8, 8]	2	1	0
7	-1.3390	0.5635	-0.0516	C	6	[4, 9, 9, -1]	1	1	0
8	0.7510	1.8034	-0.0566	C	6	[6, 6, 9, -1]	1	1	0
9	-0.6530	1.7893	-0.0521	C	6	[7, 7, 8, -1]	1	1	0
10	-1.3814	-1.8647	-0.0492	C	6	[2, 3, 3, 4]	2	1	0

Rotation and Translation





2D

- Centroid on origin
- Furthest atom on x axis
- Normalization

3D

- Centroid on origin
- Furthest atom on x axis
- Normalization
- Furthest atom from x axis to y axis

Database & Method

ChemSpider is a free chemical structure database providing fast access to over 34 million structures, properties and associated information.

Incorporation with machine learning: **Gradient boost tree**
Boosting is an ensemble technique where new models are added to correct the errors made by existing models.

Packages & Setup

UNIVERSITY of WASHINGTON

The following modules are required for the environment.

- **Python3**
- **ChemSpiPy** provides a way to interact with ChemSpider in Python. It allows chemical searches, chemical file downloads, depiction and retrieval of chemical properties.
- **XGBoost** is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way.
- **pandas, scikit-learn** and **numpy** for utilities

Tests

UNIVERSITY of WASHINGTON

Run unit tests of each function locally.

```python
def test_get_df_database_input():
    """ Check input type"""

    try:
        data_compile.get_df_database(0.1)
        raise Exception()
    except TypeError:
        pass


def test_trim_hydrogen_input():
    """ Check input type"""

    try:
        data_compile.trim_hydrogen(1, pd.DataFrame())
        raise Exception()
    except TypeError:
        pass

    try:
        data_compile.trim_hydrogen(pd.DataFrame(), 1)
        raise Exception()
    except TypeError:
        pass


def test_atom_connect_col():
    """ Check for existed connect_to column"""
    # dataframe already have connect_to column
    [_, test_bond] = data_compile.get_df(path + '/samples/user.txt')
    test_df = data_compile.get_df_user([path + '/samples/user.txt'])
    try:
        data_compile.atom_connect(test_df, test_bond)
        raise Exception()
    except ValueError:
        pass
```

Packed with travis.yml for continuous integration.

An virtual environment is created to run self-testing in remote repository.

Convenient for regular committing by each team member.

```yaml
language: python
sudo: false

python:
  - '3.6'

install:
  - pip install -r requirements.txt
  - pip install python-coveralls
  - pip install pytest-cov
  - pip install coveralls

script:
  - pytest --cov=optimol/

after_success:
  - coverage report
  - coveralls
```

Demo

Accuracy : 55.49% (2.09%) with 10-fold cv

```
demo_input
```

	2d_x	2d_y	periodic_#_2d	bond_1_2d	bond_2_2d	bond_3_2d
0	4.6055	-1.9942	7	1	0	0
1	3.4542	-2.6574	6	2	1	0
2	3.4542	-3.9884	8	0	1	0
3	2.3028	-1.9942	6	2	1	0
4	1.1514	-2.6620	6	1	1	0
5	0.0000	-1.9942	6	1	1	0
6	0.0000	-0.6632	7	1	1	0
7	1.1514	0.0000	6	1	1	0
8	2.3028	-0.6632	6	1	1	0

```
model.predict_3d(demo_input,estimator)
```

	3d_x	3d_y	3d_z
0	1.799376	-0.046763	-0.040233
1	0.557831	-0.215743	0.335885
2	-1.711492	0.673397	0.058597
3	1.522962	-0.532426	-0.194860
4	1.560337	1.244744	-0.046162
5	0.713294	-0.253275	-0.079990
6	-0.025737	0.218338	-0.080859
7	0.784534	-0.225123	0.003144
8	0.078967	-0.438747	-0.015504

Future Work

UNIVERSITY of WASHINGTON

Visualization

Improve accuracy

Implement more elements

Implement more functions and use case