Cheminformatics Library

GHDDI-AIDD Team

Updated: May 20, 2020

Table of Contents

1. Handy Functions

- HandyFullMolecularProperties.py
    - Calculates canonical_smiles for each SMILES sequence
      transform_smiles*(inputdata, inputsmilefield='SMILES', save_csv=False, parallel=False, chiral=True)*

    - Obtains InChiKey from each SMILES sequence
      get_inchikey*(inputdata, inputfield='cleaned_smiles')*

    - Calculates all available RDKit molecular properties for each SMILES sequence
      get_molecular_properties*(inputdata, inputfield='cleaned_smiles')*

- HandyMurckoScaffoldCalculation.py
    - Calculates Murcko scaffold for each SMILES structure
      generate_scaffold*(smiles, include_chirality=False)*

2. SMILES_clean.py

- Preprocessing library to clean SMILES sequences
- Dependencies:
    - RDKit version 2019.09.3 (or above)
    - Numpy, pandas, joblib, multiprocessing
    - HandyFullMolecularProperties, HandyMurckoScaffoldCalculation
    - Main Function

Running on command line:

```
(chem) [yjin@comput7 anaconda]$ python SMILES_clean.py -file tox21.csv -input smiles
True
<class 'bool'>
start
[13:50:43] WARNING: not removing hydrogen atom without neighbors
[13:50:45] WARNING: not removing hydrogen atom without neighbors
[13:50:45] WARNING: not removing hydrogen atom without neighbors
[13:50:45] WARNING: not removing hydrogen atom without neighbors
Done
```

Output will be *_cleaned.csv

Preprocessing Functions:

smiles_preprocessing*(df, inputfield='SMILES', chiral=True, parallel=False, n_cores=-1, remove_duplicates=False, remove_error=False, remove_polymer=True, remove_inorganic=True, fps=True)*

- Function to return inputted SMILES as Cleaned SMILES, scaffolds, and Morgan Fingerprints

Inputs	
df	Pandas dataframe
inputfield	Pandas dataframe column containing SMILES strings, default='SMILES'
chiral	Keep chirality in final cleaned SMILES, default: True
parallel	Parallel processing, default: True
n_cores	Number of cores to use for parallel processing, default=-1 (Max available)
remove_duplicates	Remove duplicate cleaned SMILES from final dataframe, default: False
remove_errors	Remove error rows (NaN) from dataset, default: False
remove_polymer	Remove polymers (SMILES sequences containing '.') and keeping the largest strand, default: True
remove_inorganic	Removes inorganic SMILES sequences from dataframe, default: True
fps	Generates Morgan Fingerprints for all valid cleaned SMILES, default: True, cannot be used if parallel is True

Outputs	
cleaned_smiles	Pandas dataframe
fps1	List of Morgan Fingerprints

_smiles_process*(smiles, remover_tool, chiral=True, remove_polymer=True, remove_inorganic=True)*

- Basic unit of smiles_preprocessing. Cleans SMILES sequence input and outputs a cleaned SMILES sequence. This function can be used by itself by importing SMILES_clean into python script.

Inputs	
smiles	SMILES sequence, python str
remover_tool	Salt remover module from RDKit, loaded by default from previous step due to memory issues. Can be loaded inside function if variable is None
chiral	Keep chirality in final cleaned SMILES, default: True
remove_polymer	Remove polymers (SMILES sequences containing '.') and keeping the largest strand, default: True
remove_inorganic	Removes inorganic SMILES sequences from dataframe, default: True

Outputs	
_cleaned_smiles	Cleaned SMILES sequence, python str
_scaffolds	Murcko scaffolds SMILES sequence, python str

Search Functions

These functions inside SMILES_clean are used for dataset searching.

identify_duplicates2*(f1, f2, df_field='cleaned_smiles')*

- Identify duplicates from the 2 dataframes based on the indicated dataframe column field

Inputs	
f1	Pandas dataframe of column data to lookup
f2	Pandas dataframe of column lookup table (ex. *if f1[column][i] in list(f2[column])*)
df_field	Column from both f1 and f2 to identify duplicates, python string

Outputs	
FP	SMILES list, deprecated
idxlist	Index of duplicate on dataframe being searching
catlist	Index of duplicate on lookup dataframe

get_column_value*(df, df2, c, data_field=None)*

- After getting the identified duplicate indices, map column data_field values from df2 onto df, output df with new column of data with the name from data_field

Inputs	
df, df2	Same logic as above with f1 and f2, df is output with new column: df[data_field]
c	List of indices of duplicates mapped from df2 (and f2 above)
data_field	Column from df2 (f2) to map onto df, python string

Helper Functions

These functions inside SMILES_clean do not have functions outside the python file. They are only used to assist processes within the SMILES preprocessing main function.

- NeutralizeReactionsDict1
- NeutralizeCharges
- neutralizeRadicals
- countAtoms
- countCarbons
- _process_loop

Other Functions

- identify_duplicates3 (NO LONGER IN DEVELOPMENT)
- get_scaffold_index(df, scaffold_data=None) (ONLY USED FOR FDA/REFRAME LIBRARIES)
- M = similarity_matrix(fps)
- df_smiles_rename_columns(df)
- smiles_train_test_sets(df, inputcolumn='cleaned_smiles')

3. SMILES_similarity.py

- MCS linkage clustering function
- Dependencies:
    - RDKit version 2019.09.3 (or above)
    - Numpy, pandas, joblib
    - SMILES_clean

MCS_clustering(object)
Python class object for clustering workflow

- Initial Inputs

Inputs	
df	Dataframe or directory for csv file
shuffle	Shuffle dataframe
random_state	Pseudorandom number

- ***sim_matrix****(self, inputfield='smiles')*
    - Generates similarity matrix and cleaned SMILES dataframe for object

- ***clustering{_p}****(self, M=None, correct_smiles=None, threshold=0.9, sim_threshold=0.7, n_cores=8, output='list')*
    - _p is parallel version; non-parallel function does not contain n_cores input

Inputs	
M, correct_smiles	Default None, already loaded within Python class. If running separately, indicate M similarity matrix and correct_smiles pandas dataframe containing cleaned_smiles column
threshold	Threshold for Jaccard Similarity
sim_threshold	Threshold for Tanimoto similarity matrix
n_cores	Number of cores, default=8
output	If 'list', output will be list of indices lists, if 'df' then the output will be column 'cluster_n'

Final output for MCS_clustering class is either list of cluster indices or dataframe with cluster labels.

Helper Functions
These functions inside SMILES_similarity do not have meaningful usage outside the python file. They are only used to assist processes within the main class.

- _similarity_matrix(fps) [DEPRECATED, use def from SMILES_clean instead]
- similarity_to_distance
- clustering_loop
- clustering_recount
- jaccardscore
- cluster_mapping
- cluster_new_list() [TODO], search and rank new SMILES by cluster

4. SMILES_readacross.py

read_across_class*(df, feature=None, clusters='clusters', read_across_column='read_across')*

- Simple read across using mode of one numerical feature to map onto other SMILES structures of the same cluster with missing numerical class values

read_across_label*(df, feature=None, clusters='clusters', read_across_column='read_across')*

- Simple read across using mode of one string feature to map onto other SMILES structures of the same cluster with missing labels

Inputs for both functions are the same. It is up to the user to determine which one is appropriate for their workflow.

Inputs	
df	Dataframe containing a cluster/label column and a column with numerical or string features
feature	Feature inside dataframe df to be mapped, string
clusters	Column of clusters labels inside dataframe df, string
read_across_column	Name of new column created by this function, default:'read_across', python string

Output	
df	Pandas dataframe with added df[read_across_column] column