rxnutils – A Cheminformatics Python Library for Manipulating Chemical Reaction Data

Christos Kannas¹, Amol Thakkar¹,², Esben Bjerrum¹,², Samuel Genheden¹*

¹ Molecular AI, Discovery Sciences, R&D, AstraZeneca, Gothenburg, Sweden
² Department of Chemistry, Biochemistry and Pharmaceutical Sciences, University of Bern, Freiestrasse 3, 3012 Bern, Switzerland
³ Currently at Odyssey Therapeutics, Cambridge, MA, USA
* Corresponding author: samuel.genheden@astrazeneca.com

Keywords: chemical reactions, reaction templates, USPTO

We introduce the Python package rxnutils that can be used to manipulate chemical reactions, reaction templates and reaction datasets. The package is built entirely on open-source software such as RDKit and is designed with robustness, extendibility, and reproducibility in mind. Currently, it consists of three sub-packages one for working with chemical entities, one provides pipelining capabilities, and one provides an end-to-end pipeline for preparing the US patent reaction dataset for modelling. In this software research note we discuss the design of the package and provide some code examples. The project is open-source with a Apache 2.0 license and available at GitHub: https://github.com/MolecularAI/reaction_utils

Introduction

Computer-aided synthesis planning (CASP) has seen increased research activity in the last decade due to the progress in machine learning and deep learning (Schwaller et al. 2022). CASP has the potential to impact the synthesis of chemical materials by for instance providing predictions of synthetic routes for novel compounds, the assessment of predicted reactions, and the optimization of reaction conditions (Coley et al. 2019). Underlying many such approaches is a computer model that typically takes as input a chemical system. The chemical system can for instance be a molecule, a chemical reaction, or a generalization of a chemical reaction, described using a machine-readable representation (David et al. 2020). A chemical reaction can be represented for instance by a SMILES (simplified molecular-input line-entry system) string, a difference fingerprint, or a graph. To process such representations, one or more packages tailor-made for working with chemical entities are used.

There are many software libraries for working with chemical structures, datafiles, and databases, such as the Chemistry Development Kit (Willighagen et al. 2017) and OpenBabel (O’Boyle et al. 2001). RDKit¹ is particularly popular in CASP because it is open-source and provides a Python interface. The toolkit provides general routines for working with for instance molecules, reactions, and chemical queries. A toolkit particularly interesting for CASP is the CGRtools package (Nugmanov et al. 2019), which provides an alternative graph representation of the reaction, the condensed graph of reaction (CGR; Varnek et al. 2005). The toolkit provides routines for manipulating the CGR, such as template extraction, atom-mapping comparison, and reaction balancing. In the present work, we will introduce a high-level library, which we simply call rxnutils, which can be used for working with reactions in various CASP applications. We decided to build it on top of RDKit because many tools and software

¹ https://github.com/rdkit/rdkit

19 August 2022
uses, e.g. SMILES and SMARTS (SMILES arbitrary target specification) for handling chemical information, and RDKit is one of the most widely used packages to handle such information.

Historical reaction data are readily available for modeling; datasets like the extractions from US patent office (USPTO) (Lowe, 2017) and Open reaction database (Coley et al. 2020) are publicly available, although they are typically smaller than proprietary datasets such as Reaxys\(^2\), Pista\(^3\) and CAS\(^4\). Working with such datasets, which can contain tens of millions of chemical reactions, using a cheminformatics toolkit like RDKit requires efficient, robust, and reproducible pipelines. Unfortunately, it is not common to publish the code of such pipelines. The rxnutils package includes a framework for creating data pipelines, solving the issue of efficiency and robustness. Furthermore, the rxnutils package introduce a complete set of pipelines for end-to-end preparation of the USPTO dataset, starting by downloading the raw datafile and performing transparent and reproducible steps to clean, validate and prepare the reaction data for modeling. This solves the reproducibility issue of existing datasets.

We release rxnutils open source with the intention of contributing to transparent and reproducible protocols for working with reaction data that underlies our internal modeling efforts. However, we envisage the practitioners in the field realize the benefit of such protocols and start to adopt them for their work, thereby paving the way for a sustainable research community on CASP.

![Figure 1](image)

**Figure 1 – Package overview. The rxnutils package consists of three subpackage: chem, pipeline and data.uspto.**

The rxnutils package

The rxnutils package is written in Python 3 and depends exclusively on open-source packages, mainly RDKit. It currently consists of three sub-packages. The first sub-package is used to work with chemical reactions and reaction templates. The second sub-package is a lightweight and simple data pipeline to manipulate reaction datasets. The third sub-package is a pipeline for downloading, transforming and atom-map reaction data from the US patent office (USPTO). We will now explain these sub-packages

\(^2\) https://www.reaxys.com/
\(^3\) https://www.nextmovesoftware.com/pistachio.html
\(^4\) https://www.cas.org/about/cas-content
in more detail and provide code examples for key applications. An overview of the package is provided in Figure 1.

Chemical reactions and reaction templates

Chemical reactions are represented as reaction SMILES, where SMILES of the reactants, reagents, and products are separated with a “>” character. Each component of these SMILES is separated by a “,” and rxnutils also supports grouping components using parenthesis. Typically, the reaction SMILES is atom-mapped, i.e., each atom in the SMILES string is augmented with an atom-map number that informs the correspondence between the reactants and products. We encapsulate a chemical reaction in a class that provides some simple curations and cleaning upon instantiation. Redundant atom-mapping on reactants is removed, reagents and reactants are clearly separated, molecules are neutralized, and are sanitized with RDKit. In Code Box 1 we show the usage of the ChemicalReaction class on a Bechamp reduction taken from the USPO dataset.

```python
from rxnutils.chem.reaction import ChemicalReaction
"[Cl:1][C:2][N:3]=[CH:4][C:5][C:10][[CH:11]=1]=C:9([NH2:12])[CH:8]=C:7(C:6)=2"
)
rxn = ChemicalReaction(reaction)

rxn.reactants_list
>>> ['O=[N+:12][[O-]=0]=C:9(C:8)=C:7(C:6)=2(C:4)[N:3][C:2][([Cl:1])][CH:11][C:10][21]']

rxn.agents_list
>>> ['CC(=O)O', '[Fe]', 'O', '[Na+]', '[OH]']
```

A reaction template is a generalization of a chemical reaction, indicating the reaction center, its neighbors, and other important groups as a reaction SMARTS (SMILES arbitrary target specification) string. The rxnutils package can extract templates using routines from the RDChiral package (Coley et al. 2019), as first outlined in Thakkar et al. (Thakkar et al. 2020). We also introduce novel approaches to generate a fingerprint and hash strings for the templates. RDKit produces extended connectivity fingerprints (ECFP; Rogers et al. 2010) by operating on a molecular graph that is initialized by parsing a SMILES or SMARTS string. However, because the SMARTS string in a reaction template is a concatenation of different atomic queries, not all of this information is embedded in the molecular graph. Therefore, fingerprints are produced that do not reflect, e.g., the aromaticity, the number of direct neighbors, or the formal charge of the atoms. The rxnutils package can generate an ECPF for a reaction template, by collecting atom invariants that reflect all the information in the SMARTS string and then feeding these invariants to RDKit which produces the fingerprint. As such, the reaction SMARTS fingerprint produced captures all of the essential features of the reaction template, in the same way molecular fingerprint does for a molecule parsed from a SMILES string. Furthermore, the fingerprint bits can be used to generate a hash string, uniquely identifying a template. Code Box 2 shows how a reaction template can be generated and how fingerprints and hash strings can be computed.
**Code Box 2:** Example of creating and using reaction template. The `generate_reaction_template` method will produce both a forward and a retro template that are encapsulated in ReactionTemplate objects. These objects can be used to generate fingerprints and hash strings for the templates.

```python
rxn.generate_reaction_template(radius=1)
rxn.retro_template.smarts
>> '[NH2;D1:+0:1]-[c:2]>>O=[N+;H0;D3:1][-[O-]}-[c:2]'

rxn.retro_template.fingerprint_bits(radius=2)
>> defaultdict(int,
   {2786059672: 1, 301684112: 1, 126715431: 0, 300044842: -1, 185773645: -1, 542960973: -1, 3092111213: -1, 1215786640: -1, 3015400277: -1, 2450987999: -1})

rxn.retro_template.hash_from_bits()
>> '664f7756f7526ad6c07e82dd3b3c15f7aabccbc99837dd221f05068a2cfc385b5'

rxn.retro_template.hash_from_smarts()
>> 'bf7fa4d8f5b318dd5774d08ea0b729855570261ec03755bc563ec'
```

### Data pipelines

If a collection of (atom-mapped) reaction SMILES are provided as a comma-separated flat file, the rxnutils package provide simple pipelining capabilities. These features can be used to chain together simple actions on the reaction SMILES, such as removing bad SMILES strings, removing atom-mapping from reagents, and counting the number of components in the reactions. The user can create a pipeline by specifying such actions in a configuration file together with input arguments, and then apply this pipeline on their datafile. In version 1.0.0 of rxnutils, 33 actions are implemented, and we provide guides how to straightforwardly implement novel actions. To increase pipeline throughput, simple parallelization is provided by the Swifter package, \(^5\) or the pipeline can be applied to a subset of the input file, making it possible to perform custom and more efficient parallelization. In Code Box 3 we show a simple pipeline that first removes unsanitizable molecules and then move all reagents to the reactant side of the reaction SMILES.

**Code Box 3:** Example bash commands for creating and executing a data pipeline. The pipeline specification is in YAML format, and the example shows two actions. The first action will remove unsanitizable molecules and the second will move all reagents to the reactant side of the reaction SMILES.

```bash
cat << EOF > pipeline.yml
remove_unsanitizable:
  in_column: ReactionSmilesClean
  out_column: ReactionSmilesClean
reagents2reactants:
  in_column: ReactionSmilesClean
  out_column: ReactionSmilesClean
EOF

python -m rxnutils.pipeline.runner --pipeline pipeline.yml --data uspto_data.csv --output uspto_cleaned.csv
```

---

\(^5\) https://github.com/jmcarpenter2/swifter
USPTO data

Having access to open-source modeling-ready reaction data is paramount to any CASP method and the reaction data extracted from the US patent office has become the de-facto open-source standard for benchmarking CASP methods. Several curated, filtered and cleaned versions of this dataset have been provided and open-sourced with several publications (Schneider et al. 2016, Jin et al. 2017, Lin et al. 2022). However, it is for instance not possible, or at least very difficult, to reproduce the various subsets of the dataset in use because the code used to generate those datasets was never published. In addition, the original USPTO dataset was mapped with an atom-mapper that since has shown to have rather low performance (Lin et al. 2022), suggesting that redoing the atom-mapping could be of benefit for any subsequent modeling with this dataset. To our knowledge, there does not exist a complete end-to-end pipeline that downloads the USPTO sources and produces an atom-mapped, modeling-ready version of the data. The rxnutils package provides such a pipeline that is built entirely upon freely available, open-source packages. Downloading and cleaning of the data are done with rxnutils routines, whereas atom-mapping is produced with the rxnmapper package (Schwaller et al. 2021). The pipeline starts from the two provided flat-files with reaction SMILES and it then performs basic cleaning and filtering of the data necessary to use the rxnmapper package; unsanitizable molecules are removed, reactions without any reactants or products are removed as well as reactions with more than a total of 200 atoms in both reactants and products. Of the 3,748,191 reactions in the original USPTO datasets, 3,740,597 were kept for the atom-mapping. The produced datafile from this pipeline is modeling-ready, but of course, it is possible to further curate, filter, and clean the data because there exists potentially a lot of poor data.

Comparison to existing frameworks and projects

Both RDKit and CGRTools can be used to manipulate reactions as they provide basic routines for manipulating chemical structures. However, they are low-level toolkits that provide a set of routines to manipulate chemical graphs in order to for instance standardize structures, assign reaction roles, and apply reaction templates. Thereby, a user of such a library needs to chain together a number of routines in order to perform routine tasks. The rxnutils package is a high-level framework that leverages the power of RDKit and other low-level packages to provide an easy-to-use interface to commonly task in CASP.

Furthermore, the rxnutils package is the first package to our knowledge that provides simple pipelining capabilities to work with reaction data and the first package to provide an end-to-end pipeline for preparing the USPTO dataset. The publication presenting the rxnmapper tool (Schwaller et al. 2021) released a re-mapped USPTO dataset, and a recent comparison between different software for atom-mapping (Lin et al. 2022) also released a new, curated and atom-mapped (with rxnmapper) version of the USPTO data. However, both of these releases did not provide an end-to-end pipeline that could be transparently and reproducible executed. This is important as both standardization procedures and atom-mapping software is likely to change with time, as consensus changes in the CASP community.

Conclusion

The rxnutils cheminformatics package is introduced to work with reaction data. It is a collection of routines that we have used as the basis for a variety of software internally and that we now are releasing as open-source. We hope that by providing this package as open-source we will contribute to a reproducible and sustainable community around computer-assisted synthesis planning.
Acknowledgements

We acknowledge Tomas Bastys and Thierry Kogej for discussing algorithms and bugfixes to the rxnutils package. Ola Engkvist and Alexey Voronov are acknowledged with proof-reading the manuscript.

Conflicts of Interests

The authors declare no conflicts of interests.

References


