oximachinerunner

Release v1.4.0+2.gd3edf20.dirty

Kevin Maik Jablonka, Daniele Ongari, Mohamad Moosavi, Berend

Aug 18, 2021

CONTENTS

This package can be used to estimate the oxidation states of metals in a structure using a machine learning model. This model was developed to predict the oxidation states of metal cations in MOFs.

Technical details and case studies are discussed in our preprint

Jablonka, Kevin Maik; Ongari, Daniele; Moosavi, Seyed Mohamad; Smit, Berend (2020): Using Collective Knowledge to Assign Oxidation States. ChemRxiv. Preprint. https://doi.org/10.26434/chemrxiv.11604129.v1

CHAPTER

ONE

CONTENTS

1.1 Getting Started

1.1.1 Installation

We recommend installing oximachinerunner in a clean virtual environment environment (e.g., a conda environment). On macOS you need to run *brew install libomp* first to enable multithreading for the *XGBoost* library.

You can install the latest stable release from PyPi using

```
pip install oximachinerunner
```

or the latest development version using

```
pip install git+https://github.com/kjappelbaum/oximachinerunner.git
```

Note that the installation will take significant (>500 MB) hard drive storage as the models contain a k-nearest neighbors esimator that basically stores the complete training set.

Some parts of the code are accelerated using just-in-time compilation (jit) using numba. This can benefit from threading layers. You can enable this using `pip install tbb`. If you do not do so, you might see warnings like `The TBB threading layer requires TBB version 2019.5 or later`.

1.1.2 Running Inference

The most common use case for oximachinerunner is to estimate the oxidation states for metal cation in a crystal structure. To do so, you only need the following lines of code

```
from oximachinerunner import OximachineRunner

oximachine_instance = OximachineRunner()
results = oximachine_instance.run_oximachine(<structure>)
```

In the code snippet above, `structure` can be a `pymatgen.Structure`, `ase.Atoms` object or a filepath to a *cif*. The output of the run will be a *OrderedDict* with:

- A list of oxidation state predictions
- A list of indices of the metal sites
- Strings indicating the metal
- The predictions of the base estimators
- The estimated probabilites

1.1.3 Models

When you create an instance of the `OximachineRunner` class you can choose which model you want to use. Currently, we provide a model that was trained only on MOFs and another one that was trained on all chemistry deposited in the Cambridge Crystallographic Database (CSD) plus additional structures from the Crystallographic Open Database (COD). By default, oximachinerunner uses latter model. You can list all availabel models using the `available_models` attribute.

If you use the package for the first time, it will automatically download the models. You can turn this behavior off by setting `automatic_download=False` in the class initialization.

1.1.4 Exceptions

- *No metal in structure*: *OximachineRunner* can only be used to predict oxidation states of metals. If a input structure does not contain a metal, we will return a empty dictionary.
- *Parsing error*: Internally, we will convert all inputs into a pymatgen structure object. This only works if pymatgen can parse the structure. Note that a pymatgen Structure is a periodic object, hence we need some information about the cell. Also, there might be issues in case the *CIF* is formatted in a way pymatgen cannot handle (e.g., using *_atom_site_Cartn_x* instead of *_atom_site_fract_x*). In all these cases, we will raise a `ParsingError` execption.
- *Featurization error*: In some cases the featurization might fail. Then, we will raise a `FeaturizationError` exception.
- *Prediction error*: In some cases the prediction might fail. Then, we will rasie a `PredictionError` execption.

All errors are subclasses of `OximachineRunnerException`, which you can catch in order to handle all of the aforementioned error types in one go if desired.

1.1.5 Graphical user interface

If you want to have a graphical user interface for this application, you can use the oximachinetool. You can try this application online.

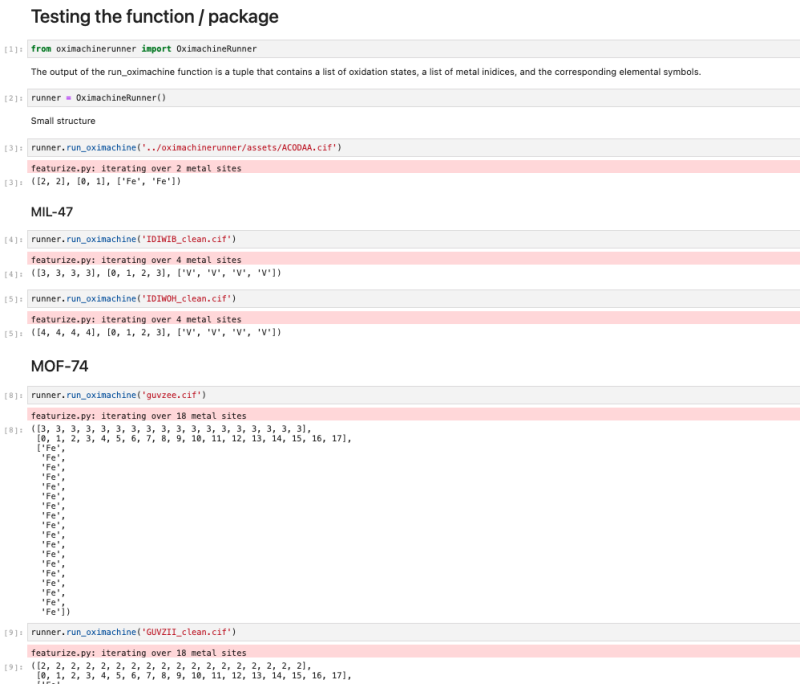
1.1.6 Caveats

This model works excellent on a test set but it might give fully unphysical predictions in some cases when it is used outside the domain of applicability (structures similar to the ones in the MOF subset of the CSD). This is currently estimated with an uncertainty vote.

1.2 Tutorials

1.2.1 Using oximachinerunner to replicate the case study shown in the main text

Click on the image to open the notebook in mybinder.



Testing the function / package

```
[1]: from oximachinerunner import OximachineRunner
```

The output of the run_oximachine function is a tuple that contains a list of oxidation states, a list of metal inidices, and the corresponding elemental symbols.

```
[2]: runner = OximachineRunner()
```

Small structure

```
[3]: runner.run_oximachine('../oximachinerunner/assets/ACODAA.cif')

featurize.py: iterating over 2 metal sites
[3]: ([2, 2], [0, 1], ['Fe', 'Fe'])
```

MIL-47

```
[4]: runner.run_oximachine('IDIWIB_clean.cif')

featurize.py: iterating over 4 metal sites
[4]: ([3, 3, 3, 3], [0, 1, 2, 3], ['V', 'V', 'V', 'V'])

[5]: runner.run_oximachine('IDIWOH_clean.cif')

featurize.py: iterating over 4 metal sites
[5]: ([4, 4, 4, 4], [0, 1, 2, 3], ['V', 'V', 'V', 'V'])
```

MOF-74

```
[8]: runner.run_oximachine('guvzee.cif')

featurize.py: iterating over 18 metal sites
[8]: ([3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3],
      [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17],
      ['Fe',
       'Fe',
       'Fe',
       'Fe',
       'Fe',
       'Fe',
       'Fe',
       'Fe',
       'Fe',
       'Fe',
       'Fe',
       'Fe',
       'Fe',
       'Fe',
       'Fe',
       'Fe',
       'Fe',
       'Fe'])

[9]: runner.run_oximachine('GUVZII_clean.cif')

featurize.py: iterating over 18 metal sites
[9]: ([2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2],
      [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17],
```

1.3 Oximachinerunner API reference

Implements methods to use oximachine as part of a Python package

class oximachinerunner.**OximachineRunner**(*modelname='all'*, *automatic_download=True*)

Bases: object

Loads a model and then runs the prediction

__init__(*modelname='all'*, *automatic_download=True*)

Parameters

- **modelname** (*str, optional*) – [description]. Defaults to 'all'. Use it to specifiy a model. You can view all available models with the .available_models property
- **automatic_download** (*bool, optional*) – [description]. Defaults to True.

__repr__()

Return repr(self).

__weakref__

list of weak references to the object (if defined)

property available_models

List all the available models.

Return type List[str]

property default_mapping

Return the default mapping between model name and filename

Return type Dict[str, str]

property feature_names

Get a list of feature names

Return type List[str]

property featureset
Return the list of feature names

Return type List[str]

load_model()
Load the model and populate the namespace with the model objects.

property model
Return the model object with *.predict* method

run_oximachine(*structure*)
Runs oximachine after attempting to guess what structure is

Parameters

- **structure**(*Union[str, os.PathLike, Structure, Atoms]*) –
- **be a pymatgen.Structure**(*can*) –
- **or a filepath as str or**(*ase.Atoms*) –
- **os.PathLike** –
- **we then attempt to parse with pymatgen.**(*which*) –

Raises

- ***ParsingError*** – In case the format of structure is not implemented or in case we cannot convert the input into a pymatgen Structure object.
- ***FeaturizationError*** – In case the featurization fails.
- ***PredictionError*** – In case the prediction fails.

Returns

with the keys metal_indices, metal_symbols, prediction, max_probas, base_predictions

Return type OrderedDict

property scaler
Return the scaler object with *.transform* method

Some general utility functions for the oxidation state mining project

class oximachinerunner.utils.**SymbolNameDict**
Bases: object

Parses the periodic table json and returns a dictionary with symbol: longname

__init__()
Initialize self. See help(type(self)) for accurate signature.

__weakref__
list of weak references to the object (if defined)

get_symbol_name_dict(*only_metal=True*)
Iterates over keys and returns the symbol: name dict.

oximachinerunner.utils.**cbk_for_urlretrieve**(*a*, *b*, *c*)
Callback function for showing process

oximachinerunner.utils.**diff_to_18e**(*nvalence*)
The number of electrons to donate to achieve 18 electrons might be an interesting descriptor, though there are more stable electron configurations

oximachinerunner.utils.**download_all**()
Download model and scaler

oximachinerunner.utils.**download_model**(*url*, *destination*, *md5*)

Downloads file from url to destination and checks md5 hash

Parameters

- **url** (*str*) – URL
- **destination** (*Union[Path, str]*) – Path to which the downloaded file will be saved
- **md5** (*str*) – Expected md5 hash

Raises

- **Exception** – [description]
- **Exception** – [description]

oximachinerunner.utils.**flatten**(*items*)
Yield items from any nested iterable; see Reference.

oximachinerunner.utils.**has_metal_sites**(*structure*)
Returns True if there is a metal in the structure.

oximachinerunner.utils.**md5sum**(*filename*)
Gets the md5 hash of a file

oximachinerunner.utils.**model_exists**(*path*, *md5*)
Checks whether a model if the expected md5 hash exists at the path

oximachinerunner.utils.**read_pickle**(*filepath*)
Does what it says. Nothing more and nothing less. Takes a pickle file path and unpickles it

Defining custom exceptions for OximachineRunner

exception oximachinerunner.errors.**FeaturizationError**
Bases: *oximachinerunner.errors.OximachineRunnerException*

Error that is thrown if the featurization fails

exception oximachinerunner.errors.**ModelNotFoundError**
Bases: *oximachinerunner.errors.OximachineRunnerException*

Error that is thrown if the model could not be found

exception oximachinerunner.errors.**NoMetalError**
Bases: *oximachinerunner.errors.OximachineRunnerException*

Error that is thrown if there is no metal in the structure

exception oximachinerunner.errors.**OximachineRunnerException**
Bases: Exception

General class for oximachine errors

__weakref__
list of weak references to the object (if defined)

exception oximachinerunner.errors.**ParsingError**
Bases: *oximachinerunner.errors.OximachineRunnerException*

Error that is thrown if we cannot convert the structure into a pymatgen Structure object

exception oximachinerunner.errors.**PredictionError**
Bases: *oximachinerunner.errors.OximachineRunnerException*

Error that is thrown if the prediction fails

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

O

INDEX

Symbols

A

C

D

F

G

H

L

M

N

O

P

R

S