Entrezpy:A Python library to dynamically interact with the NCBI Entrez databases

Jan P. Buchmann and Edward C. Holmes

Marie Bashir Institute for Infectious Diseases and Biosecurity, Charles Perkins Centre, School of Life and Environmental Sciences and Sydney Medical School, The University of Sydney, Sydney, NSW 2006, Australia

2018-12-04

1 Abstract

Summary: Entrezpy is a Python library that automates the querying and downloading of data from the Entrez databases at NCBI by interacting with E-Utilities. Entrezpy implements complex queries analogous to piping commands on UNIX systems, enabling it to implement interactions with all Entrez databases as part of an analysis pipeline, which can also be adjusted on-the-fly. Entrezpy's design allows its easy extension and adjustment to the existing E-Utility functions without changes in the underlying mechanisms.

Availability and Implementation: Entrezpy is implemented in Python 3 ($\geq$3.6) and depends only on the standard library. Its source code is available at https://gitlab.com/ncbipy/entrezpy.git and is licensed under the LGPL.

Contact: jan.buchmann@sydney.edu.au

Supplementary information: https://gitlab.com/ncbipy/entrezpy/wikis/entrezpy

2 Introduction

The increasing availability of biological data has not only resulted in a multitude of genome sequence data, but also substantial increases in the amount of accompanying metadata, including phylogenies, sampling conditions, and gene ontologies. These data can also be divided into several parts: for example, genome sequences are stored as individual chromosomes, while transcriptional experiments are split into several sequencing runs according to the specific questions addressed. To consolidate this burgeoning information, sequence repositories have developed specific databases that store and link this wealth of data. The National Center for Biotechnology Information (NCBI) is one of the largest such repositories and both developed and maintains the Entrez databases that currently encompass 37 individual databases storing 2.1 billion records related to the life sciences (NCBI Resource Coordinators, 2016). NCBI offers two approaches to interact programmatically with its Entrez databases:

Entrez Direct is a powerful Perl program allowing ad hoc access to the NCBI databases through a command line interface ((Kans, 2018) , Figure 1b). It can be incorporated into Bash scripts or invoked within pipelines as an external command. However, it is not designed as a library and therefore difficult to integrate into analysis pipelines.

E-utilities (`http://eutils.ncbi.nlm.nih.gov/`) are a set of tools that allow the user to query and retrieve NCBI data using specific Uniform Resource Identifiers (URIs). NCBI data can be accessed using an URI describing the function and its parameter, such as searching a database with a specific term.

Herein, we present `Entrezpy`. To our knowledge this is the first Python library to offer the same functionalities as Entrez Direct. Existing libraries, such as Biopython (Cock *et al.*, 2009) or ETE 3 (Huerta-Cepas *et al.*, 2016), offer either a basic or a very narrow approach to interact with E-utilities. Biopython does not handle whole queries, leaving the user to implement the logic to fetch large requests, while ETE represents a library focusing only on phylogenetics. In contrast, `Entrezpy` is specifically designed to interact with E-Utilities and automatically configures itself to retrieve large data sets. Its design, the smaller code base, and that it only depends on the Python standard library, make `Entrezpy` easier to install, maintain, adjust and extend.

`Entrezpy` includes the helper class, termed Wally, to facilitate the creation and execution of complex queries and reusing previously obtained results. This allows the user to query the Entrez database and retrieve the corresponding data as part of an analysis pipeline. `Entrezpy` is licensed under the GNU Lesser General Public License and can be obtained from `https://gitlab.com/ncbipy/entrezpy`. It is documented using Doxygen and a wiki (`https://gitlab.com/ncbipy/entrezpy/wikis/entrezpy`) which has more complex usage examples.

3 Implementation

Data sets within an Entrez database are identified by their identification number. The Entrez documentation refers to this number interchangeably as either UID or ID. For the remainder of this article we will use the term UID to refer to a data set identification number. UIDs are unique within an Entrez database but not across Entrez databases.

`Entrezpy` is a library of Python classes implementing the specific steps required to interact with the E- Utilities. Querying and downloading data via the E-Utility is achieved by sending queries encoded as an URI for the specific function and the corresponding parameters. For example, the following URI searches the nucleotide database for viral genomes and returns the UIDs detected: `https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?db=nucleotide&term=viruses[orgn]`.

The E-Utility returns a response describing the search result. This includes the number of data sets recovered within the requested database and the UID sequence identifiers. To fetch this data set, an E-Utility URI must be assembled. The following E-Utility URI fetches the first four sequences from the previous query in the FASTA format: `https://eutils.ncbi.nlm.nih.gov/entrez/`

eutils/efetch.fcgi?db=nucleotide&id=1509580163,1509580026,1509580024,1509580022&rettype=fasta&retmode=text.

Entrezpy supports the E-Utilities EFetch, ESearch, ELink, ESummary, and EPost. The E-Utilities ESpell (spelling suggestions), EInfo (database statistics), ECitMatch (batch citation searching in PubMed) and EGQuery (global ESearch) are currently not supported since they can be either assembled using existing functions or have a very broad usage. Entrezpy is not primarily intended to replace an NCBI website search, but to run queries for a specific problem. The Entrezpy functions implemented use the same parameters as those described in the Entrez manual. ESearch, ELink and EPost E-Utilities can use the Entrez History server which includes a reference for the query as part of the result, consisting of the WebEnv and QueryKey values. Together, these values can be used in subsequent queries to reference a prior query and are recognized by Entrezpy.

NCBI limits query and retrieval sizes. For example, downloading summaries in JSON format is limited to 500 summaries at a time. In such cases, queries must be split into several requests to obtain the whole requested data set. Entrezpy automates these steps, enabling the easy assembly of complex E-Utility queries to search the Entrez databases and download data sets. Further, Entrez History server responses can be used to link queries, analogous to piping commands on UNIX systems and the Entrez Direct tool (Figure 1). Entrezpy is designed to analyze the response from each request as soon as it is received, allowing the implementation of checkpoints when handling large data sets, for example whether to resume after aborts or errors. We implemented multi-threading support that can be disabled since not all libraries and tools support multi-threading; that is, the SQLite3 module in the Python standard library (https://docs.python.org/3.7/library/sqlite3.html#multithreading) disallows shared connection and cursors between threads.

The class Wally simplifies assembling complex queries (Figure 1). Internally, Entrezpy assigns each query and request a unique identifier. This allows the creation of query pipelines using prior queries, thereby limiting re-downloading results to reuse the in a later step. The versatility of Entrezpy is based on the use of virtual functions and modular design. We implemented a default analyzer for all E-Utilities except for EFetch. This is deliberate, since an analyzer for an EFetch request is usually the last step in query. Given the numerous possibilities, databases and formats available, this step is best left to the pipeline developer. Creating a specific analyzer requires the implementation of only two virtual functions of the Entrezpy analyzer base class, specifically the methods to handle errors and the result. Therefore, a new and highly specific analyzer for a specific data set can be written without the need to adjust the whole request process.

Entrezpy has been designed "to do one thing and do it well". It facilitates the querying and downloading data from the Entrez databases, one of the largest life sciences data repositories, while giving a developer the freedom to easily integrate specific analysis functions.

4 Funding

ECH is funded by an ARC Australian Laureate Fellowship (FL170100022).

```
esearch -db gene -query "tp53[preferred␣symbol]␣AND␣human[orgn]"  \|
elink -target protein  | esummary  \ |
xtract -pattern DocumentSummary -element Caption SourceDb
```

(a) Entrez Direct example to fetch the 'caption' and 'source database' information for sequences in the protein database linked from results in the gene database

```python
import esearch.esearcher

p = {'db':'nucleotide',
     'term':'biomol␣trna',
     'field':'prop'}
es = esearch.esearcher.Esearcher('esearcher', email)
print(es.inquire(parameter=p).result.uids)
```

(b) Example of searching the Entrez databases using `Entrezpy` esearcher. The nucleotide Entrez database is queried for UIDs with the term 'biomo trna' in their property field. The found UIDs are printed to the standard output using the default EsearcherAnalyzer.

```python
import wally.wally

p = {'db':'gene', 'term':'tp53[preferred␣symbol]␣AND␣human[organism]'}
w = wally.wally.Wally(email)
px = w.new_pipeline()
qid = px.add_search(parameter=p)
qid = px.add_link(parameter={'db' : 'protein'}, dependency=qid)
qid = px.add_summary(dependency=qid)
analyzer = w.run(px)
for i in analyzer.result.summaries:
  print(analyzer.result.summaries[i].get('caption'),
        analyzer.result.summaries[i].get('sourcedb')))
```

(c) `Entrezpy` Wally example reproducing Figure 1a

Figure 1: `Entrezpy` usage examples. Figure 1a depicts a query using the Entrez Direct tool. Figure 1b shows the usage for a single E-Utility function, here ESearch. Figure 1c shows the same query as Figure 1a using the Wally class from `Entrezpy`. The email parameter in the `Entrezpy` examples indicates the email of the developer as required by NCBI

References

Cock, P.J.A., **Antao, T.**, **Chang, J.T.**, **Chapman, B.A.**, **Cox, C.J.**, **Dalke, A.**, **Friedberg, I.**, **Hamelryck, T.**, **Kauff, F.**, **Wilczynski, B.** *et al.* (2009) Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics (Oxford, England)*, **25**, 1422–1423.

Huerta-Cepas, J., **Serra, F.** and **Bork, P.** (2016) ETE 3: Reconstruction, Analysis, and Visualization of Phylogenomic Data. *Molecular Biology and Evolution*, **33**, 1635–1638.

Kans, J. (2018) *Entrez direct: E-utilities on the UNIX command line.* National Center for Biotechnology Information (US).

NCBI Resource Coordinators (2016) Database Resources of the National Center for Biotechnology Information. *Nucleic Acids Research*, **45**, D12–D17.