**Instructions to run the code**

1. Clone the [GitHub](https://github.com/infoqualitylab/NER-tool-assement-for-funding-organization-extraction) repository in your local computer
2. Download the contents from the [data](https://uofi.app.box.com/folder/183922985217?s=vsf6fdtgxemp72iq44mnbsy7jm0gnari) folder and update the “data folder” in your local repository with these contents.
3. Once the code and data have been downloaded in the local directory, open it using a python IDE (Preferably PyCharm). Ensure that all the codes run in the same environment.

Python package requirements –

If the below-mentioned packages are not installed in your python environment , then use the following code to install the packages.



1. pandas
2. csv
3. bs4 (BeautifulSoup)
4. re (regex)
5. math
6. os
7. sys
8. time

Application

Description automatically generated with medium confidence

**Phase1: Extraction of funding information**

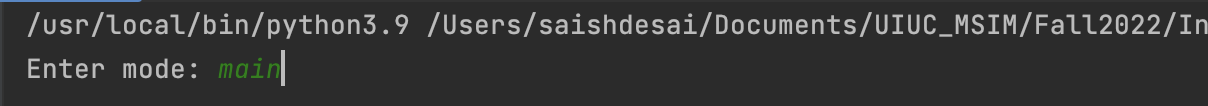
**Part A : Extracting the funding information from PubMed (Execution Time)**

1. Code -

1. main.py
2. PubMed\_API\_XML\_Retrieval.py

2. Running Instructions -

1. Run the main code.
2. Enter the mode – (main/pilot)



**Option 1 (pilot)** – This option will use the manually downloaded xml files available in the “xml\_manual” folder and generate the csv files containing the funding information

Table

Description automatically generated

**Option 2 (main)** – This option will the run the “PubMed\_API\_XML\_Retrieval.py“ module to download full text of research papers from the PMC database in PubMed and store it as an xml file.

3.Output -

The code generates an xml file containing full text of research papers and extracts the funding information from the relevant xml tags withing each file. One “.csv” will be generated for each “.xml” file. For the option 2 (main) every xml is deleted after its use and the result is stored in a “.csv” file.

Background pattern

Description automatically generated



The folder “ack\_data” stores all the resulting csv files. Each csv file contains funding information pertaining to 100 research papers.

The funding information for each paper comes from 4 types of xml tags and data for each type will be stored in separate columns namely – ACK, ACKNOWLEDGEMENT, FUNDING, FOOTNOTES

Application, table, Excel

Description automatically generated

**Note – The code “main.py” might fail to retrieve full text of all the research papers in the selected time period. I am still working fixing this error. This might happen due to excess request sent for retrieval of full text. If it fails, you can directly use the “cumulative\_ack\_data.csv” file from Part B.**

**Part B: Combing the funding information for all the research into a single “.csv” file**

1.Code -

ack\_data\_integration.ipynb

2.Running instructions –

1. Ensure that all the “.csv” files are generated in the “ack\_data” folder

Open the code “ack\_data\_integration.ipynb” using a Jupyter notebook.

1. Run all the cells in the notebook.

Background pattern

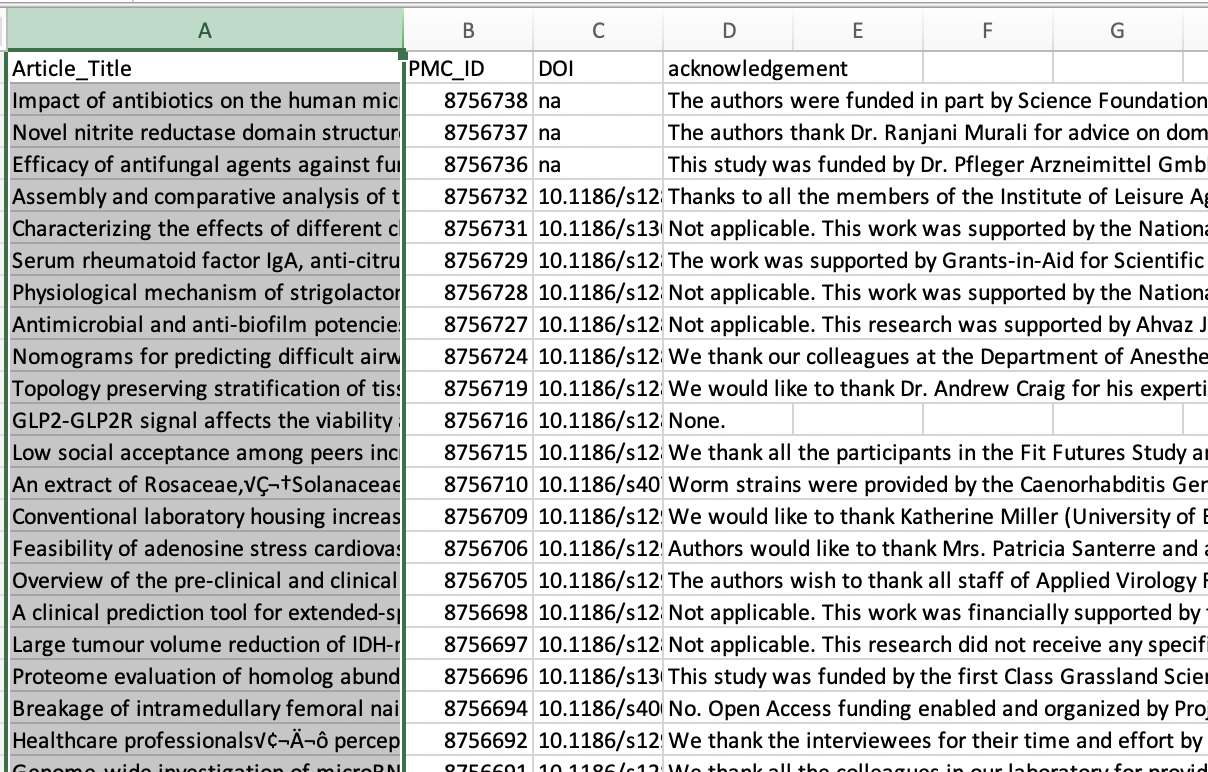
Description automatically generated

3.Output-

The output will be generated as a “.csv” . Funding information of all the research will be combined in this “.csv” . The funding information is combined in a single column named “acknowledgement”

Table

Description automatically generated with low confidence



**Phase2: Application in the selected NER tools on the funding information (Approximate Execution Time – 1.5 hours)**

1.Code – Named\_Entity\_Recog.ipynb

This code will need GPU support for running. Hence, it should be run on **Google Colab.** Update the Note Setting to GPU

Graphical user interface, text, application, email

Description automatically generated

The code uses 4 Named Entity Recognition (NER) tools for extraction of entities –

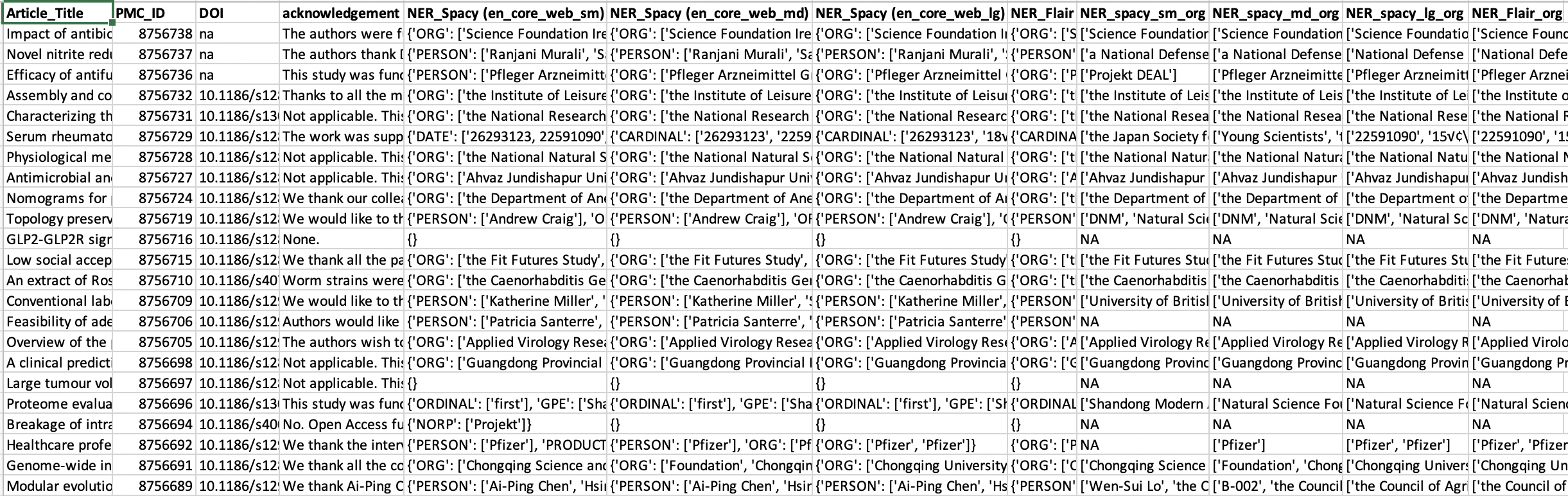
1. Spacy EntityRecognizer (en\_core\_web\_sm**)**
2. Spacy EntityRecognizer (en\_core\_web\_md**)**
3. Spacy EntityRecognizer (en\_core\_web\_lg**)**
4. Flair - English NER in Flair (large model)

2.Running instructions -

1. Upload Jupyter notebook “Named\_Entity\_Recog.ipynb” and input file “cumulative\_ack\_data.csv” on google drive. Ensure that the code and input data are in the same directory.
2. Open the Jupyter notebook and run all the cells.

3.Output

The code will generate a “.csv” and a “.pickle”. Each of these files will contain all the entities extracted using all the selected NER tools. The ack\_ner.pickle has to be downloaded from drive to the “data” folder in local repository



**Phase3:**

**Part A: Result Matching (Approximate Execution Time – 30 minutes)**

1.Code –

1. Result\_matching.ipynb

This code (Jupyter notebook) will compare the organization names identified by the NER tools with a reference list of organizations extracted from the Crossref Funder Registry

1. Crossref\_funding\_organization\_extraction\_dict\_creation.py

This code is imported as a module in the notebook. It extracts all the organizations names listed in “.rdf” file downloaded from [Crossref Funder Registry](https://gitlab.com/crossref/open_funder_registry/-/blob/master/registry.rdf).

2.Running Instructions –

1. Ensure that the files “registry.rdf” and “ack\_ner.pickle” are in the data folder
2. Open “Result\_matching.ipynb” as a Jupyter notebook and run all the cells of the notebook.

3.Output –

1. The code will compare the results from the Phase 2 (NER Model) with the organization names extracted using “Crossref\_funding\_organization\_extraction\_dict\_creation.py” (Crossref Model)
2. Text processing will be applied on organization names from the NER model and Crossref Model.
3. Based on the text processing functions 13 types of comparison (Model) will take place

|  |
| --- |
| **Model** |
| Crossref baseline vs NER baseline |
| Crossref baseline vs NER improved version I |
| Crossref baseline vs NER improved version II |
| Crossref baseline vs NER improved version III |
| Crossref baseline vs NER improved version IV |
| NER baseline vs Crossref improved version I |
| NER baseline vs Crossref improved version II |
| NER baseline vs Crossref improved version III |
| NER baseline vs Crossref improved version IV |
| NER improved version I vs Crossref improved version I |
| NER improved version II vs Crossref improved version II |
| NER improved version III vs Crossref improved version III |
| NER improved version IV vs Crossref improved version IV |

For each model output from each NER tool will be compared with the output from Crossref Funder Registry. This data will be stored in “Result\_Match.csv”

**Observation for each model**

1. Every NER tool will have a separate count (**count**) for the number of organizations identified.
2. Similarly, there will be a separate count (**match count**) for the number of organizations matching between Crossref and the NER tool output.
3. The observations will be compared using percentage of **count** from the **match count**.
4. These percentage match values will be reported.

For example, consider output for “NER\_spacy\_sm\_org” tool

Graphical user interface, application, table, Excel

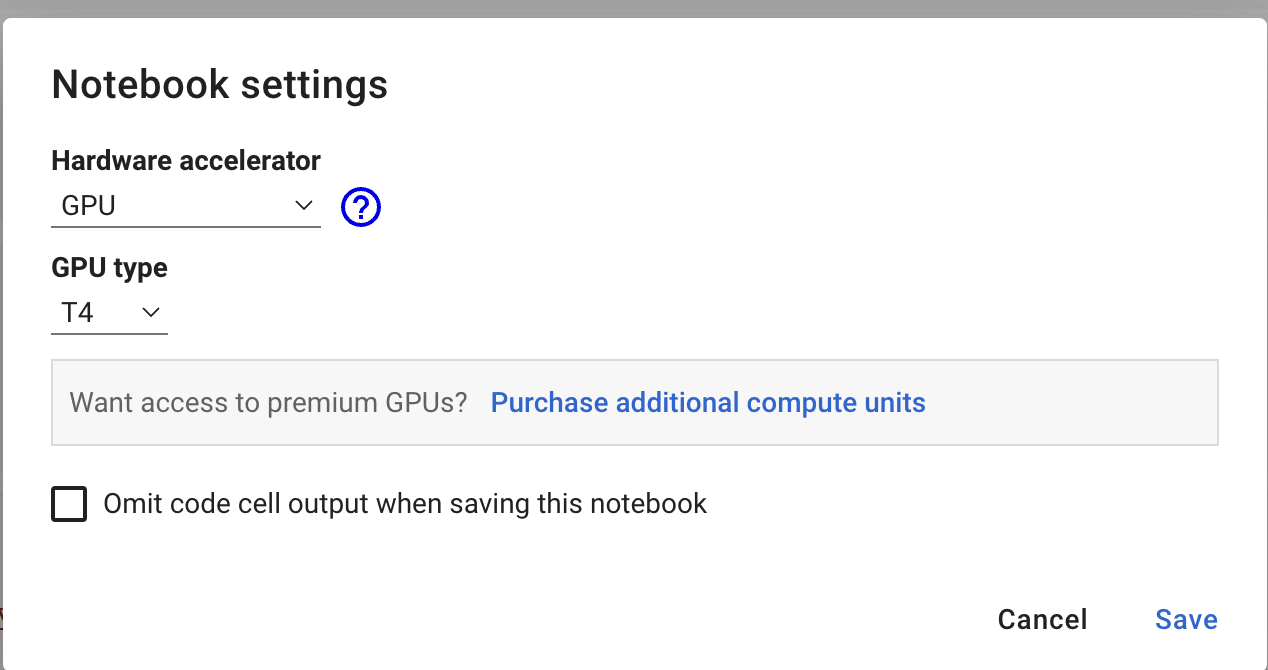
Description automatically generated

* Column B contains the count of all the organizations identified by the tool
* Column C contains the count of all matches between “Crossref” and “NER\_spacy\_sm\_org”
* Column D contains percentage match

**Part B: Applying NER on Crossref Funder Registry and categorizing cases of manual error for mismatch.**

1.Code – Apply\_NER\_on\_Crossref.ipynb

This code will need GPU support for running. Hence, it should be run on **Google Colab.** Update the Note Setting to GPU



2.Running instructions -

1. Upload Jupyter notebook “Apply\_NER\_on\_Crossref.ipynb” and input file “registry.rdf” on google drive. Ensure that the code and input data are in the same directory.
2. Open the Jupyter notebook and run all the cells.

3.Output –

The output will be a CSV file named “corssref\_ner.csv”.