PyOhdsiWebApi: A Python phenotyping library for OMOP CDM

Xujing Shen[1], Satish Kumar Anbazhagan[1], Suwarno[1], Peter Hoffmann[1]

[1]D4L data4life Asia Limited, Singapore

Background

- The aim of this work is to enable researchers to build phenotypes and cohorts in Python on OMOP [1].
- The key objective of the library is to write complex criteria in a simple query language rather than with SQL because it gets complicated and error prone as the criteria grows.
- The PyOhdsiWebApi library allows generation of patient cohorts in Python, making them available for further analysis using Numpy, PyTorch etc.
- PyOhdsiWebApi interfaces with WebAPI [2] by consuming its RESTful services.

Methods

Easy access to CDM OMOP with Python is very valuable, especially with the advent of popular Python libraries for data science and Web IDEs (e.g. Jupyter Notebook) that have made it easy for researchers to do their complete analysis in Python. The PyOhdsiWebApi library is building on the OHDSI WebAPI as a natural middleware for interacting with the OMOP database while enforcing access controls.

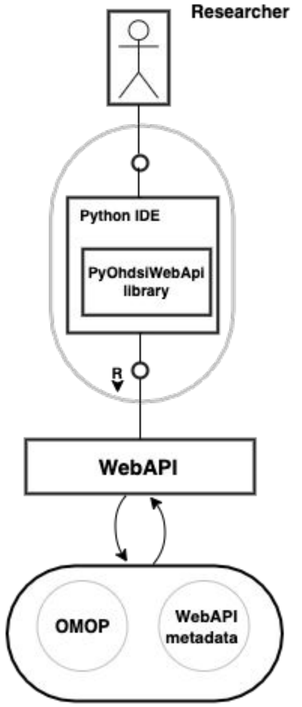

Figure 1. Client-Server Architecture

The main characteristics of the implementation are:

- PyOhdsiWebApi interacts with WebAPI to build phenotypes as per the architecture depicted in Figure 1.
- PyOhdsiWebApi translates the Python phenotype into a WebAPI JSON which is sent as a payload of the HTTP request to the WebAPI for query execution.
- The phenotypes are stored in WebAPI and generated against the OMOP CDM.
- PyOhdsiWebApi allows users to be authenticated with WebAPI either via DB or LDAP authentication.
- The design of the PyOhdsiWebApi query language is influenced by Atlas [3]. Most OHDSI entities, such as ConceptSet, ConditionOccurrence etc., are brought in.
- The sequence of defining a phenotype is also similar to Atlas, beginning with phenotype entry events, followed by inclusion criteria and exit event. Researchers who are familiar with Atlas can easily pick up PyOhdsiWebApi and start to use it.
- The phenotype data can be downloaded, to allow researchers to do future analysis.
- Once downloaded, researchers can use tools from the rich Python ecosystems, e.g. Numpy or PyTorch, for in-depth analyses.

Results

To test the the expressiveness and usability of PyOhdsiWebApi, we used it to model a set of publicly available phenotypes from the Phenotype Knowledgebase website PheKB [4]:

- We used PyOhdsiWebApi to successfully implement the ADHD phenotype algorithm, the Coronary Heart Disease Algorithm, the Type 2 Diabetes Mellitus, the Peanut Allergy, and the Sleep Apnea Phenotype.
- A phenotype of patients who are exposed to at least one dementia drug modelled using PyOhdsiWebApi is shown in Figure 2. The dementia medications are represented as a ConceptSet named `dementia_meds`, with a list of drug standard concept ids as concepts. The phenotype is a query with `dementia_meds` drug exposure as an entry event. To further filter the phenotype, the `dementia_meds` drug exposure is enforced to have occurred at least once at any time by an inclusion criteria. Finally, the phenotype is created in the WebAPI and executed against the OMOP CDM.

```
# define concept set
dementia_meds = ConceptSet('Dementia meds').set_concepts([836654, [...], 715997])
# define phenotype with name
dementia_drug_phenotype = Query('Patients who has taken a dementia drug')
     .add_initial_events([DrugExposure(dementia_meds)])
     .add_inclusion_criteria(CriteriaGroup(name='Has taken at least 1 drug in dementia_meds')
         .addDomainCriteria(
         DrugExposure(dementia_meds),
         Occurrence(HavingOccurrence.AT_LEAST,1),
         TimeWindow(CutOff.BEFORE_INDEX_DATE,-1,CutOff.AFTER_INDEX_DATE, -1))
# save & generate phenotype
dementia_drug_phenotype.save_cohort_definition()
dementia_drug_phenotype.generate_cohort()
```

Definition of the set of concept IDs for Dementia medications

Filtering of the patient who got drugs for Dementia treatment

Filtering for at least 1 drug exposure

Filtering that the drug exposures happened at any time

Figure 2. Phenotype built in PyOhdsiWebApi

Conclusions

Machine learning / Deep learning have become very important in today's research and Python undoubtedly has gained popularity in this space. However, existing OMOP query tools for Python (e.g. InspectOMOP [5]) still use SQL-like Object Relational Mapper APIs to write complicated queries. PyOhdsiWebApi offers an API that is significantly simpler to use and read, following the philosophy of existing libraries for other programming language (e.g. RohdsiWebAPI [6] for R).

PyOhdsiWebApi is able to simplify the overall work of the researchers by enabling them to define complex phenotypes in simple queries with a popular language that has a solid ecosystem of libraries. Thereby our solution creates a seamless workflow for researchers by continuing to analyze end-to-end in the Python environment without the need to switch between multiple languages or tools for different tasks. We will continue to enhance the library and add more features on the basis of the feedback given by the researchers using this library. The plan is to open source in the coming months.

Reference

[1] OMOP Common Data Model. https://www.ohdsi.org/data-standardization/the-common-data-model/
[2] OHDSI WebAPI. https://github.com/OHDSI/WebAPI/wiki
[3] OHDSI Atlas. https://www.ohdsi.org/atlas-a-unified-interface-for-the-ohdsi-tools/
[4] PheKB. https://www.phekb.org/
[5] InspectOMOP. https://github.com/jbadger3/inspectomop
[6] RohdsiWebApi. https://github.com/OHDSI/RohdsiWebApi

Contact: peter.hoffmann@data4life-asia.care

Acknowledgements

We would like to thank Dr. Benjamin S Glicksberg and Dr. Riccardo Miotto for supporting this work. We also thank Dr. Morten Ernebjerg, Dr. Matthias Steinbrecher and Dr. Péter Adorján for providing valuable inputs.