Building Machine Learning Datasets Using the EU Cellar APIs + Knowledge Graph

Introduction

To build a machine learning dataset using the Publications Office of the EU's Cellar repository, you can leverage the structured metadata available through its Knowledge Graph. The Knowledge Graph represents its data using the Work-Expression-Manifestation-Item (WEMI) hierarchy, as defined by the FRBR (Functional Requirements for Bibliographic Records) model. This hierarchy describes the relationships between different levels of a publication's representation, from abstract ideas (works) to specific digital files (items). The URL of the digital file (i.e., the document) is found at the item level. Once the file URL is extracted, the content can be downloaded via a dedicated API.

Two Cellar interfaces are needed:

- The Knowledge Graph endpoint of the Cellar is used to search/list the content files according to your criteria thanks to SPARQL queries. The knowledge graph is storing the metadata of all publications stored in Cellar. For example, you can retrieve all URLs of English PDF legislative texts.
- The API to download the digital file of a publication in a given language and a given format from its URL. The URL has the following pattern: http://publication.europa.eu/resource{domain}/{identifier). For example : http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1.0006.01/DOC_1. In this case, the domain + identifier must be known. It is the reason why the Knowledge Graph endpoint must be used before to extract these identifiers.

Prerequisites

Model of the metadata

In the Knowledge Graph, the metadata of publications are represented thnaks to a model, the CDM (Common Data Model), an ontology. This ontology defines mainly classes, as cdm:case-law, and properties as cdm:work_created_by_author. Additionally, the ontology allows to structure the publications following the WEMI (Work/Expression/Manifestation/Item) hierarchy.

The WEMI Hierarchy

1. **Work**: the abstract concept, such as a legal case or a policy document.
2. **Expression**: metadata relative of the language of the publication (e.g, English)
3. **Manifestation**: metadata relative of the format of the publication (e.g, PDF) on the language of his expression
4. **Item**: metadata about the digital file.

In this structure, the URL of the digital file is located at the **Item** level, and is essential for downloading the actual content.

Identifiers

For each publication, each level of the WEMI hierarchy has his unique Cellar URL identifier. On the Knowledge Graph, all metadata are attached to this URL. Moreover, some Production Identifier are also attached to a Cellar identifier thanks to the property owl:sameAs.

In the previsous example, the item with URI http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1.0006.01/DOC_1 and a production URI: http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01.ENG.pdfa1a.l_00120140104en00010003.pdf will have:

- A parent manifestation with Cellar URI: http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1.0006.01 and a production URI: http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01.ENG.pdfa1a
- The parent expression with Cellar URI: http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1.0006 and a production URI: http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01.ENG
- The parent work with Cellar URI: http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1 and a production URI: http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01

SPARQL Queries for Data Retrieval

You can use SPARQL queries to retrieve data from the Knowledge Graph, filtering based on criteria like language, document type, subject area, etc.

The Knowledge Graph interface is available at : https://publications.europa.eu/webapi/rdf/sparql

Below is an example query to retrieve English-language pdf documents related to European court cases between 2016 and 2020.

Example SPARQL Query

```
prefix cdm: <http://publications.europa.eu/ontology/cdm#>
prefix purl: <http://purl.org/dc/elements/1.1/>
select distinct ?item
where {
?work cdm: date_creation_legacy ?date.
?work a ?class.
?expr cdm:expression_belongs_to_work ?work ;
cdm:expression_uses_language ?lang .
?lang purl:identifier ?langCode .
?manif cdm:manifestation_manifests_expression ?expr;
cdm:manifestation_type "pdfa1a".
?item cdm:item_belongs_to_manifestation ?manif.FILTER (?date >
"2016-05-23T10:20:13+05:30"^^xsd:dateTime AND ?date < "2020-05-
23T10:20:13+05:30"^^xsd:dateTime).
FILTER(?class in
(<http://publications.europa.eu/ontology/cdm#document_cjeu>,
<http://publications.europa.eu/ontology/cdm#case-law>,
<http://publications.europa.eu/ontology/cdm#summary_case-law>,
<http://publications.europa.eu/ontology/cdm#summary_case-
law_jure>))
} LIMIT 1000
```

The result will list the item identifiers corresponding to the digital files.

Below is an example query to retrieve English-language pdf documents related to VAT rate.

Example SPARQL Query

```
prefix cdm: <http://publications.europa.eu/ontology/cdm#>
prefix purl: <http://purl.org/dc/elements/1.1/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
select distinct ?item
where {
?work cdm: date_creation_legacy ?date.
?expr cdm:expression_belongs_to_work ?work ;
cdm:expression_uses_language ?lang .
?lang purl:identifier ?langCode .
?manif cdm:manifestation_manifests_expression ?expr;
cdm:manifestation_type "pdfa1a";
?item cdm:item_belongs_to_manifestation ?manif.
FILTER (?date > "2016-05-23T10:20:13+05:30"^^xsd:dateTime AND
?date < "2020-05-23T10:20:13+05:30"^^xsd:dateTime).
?w cdm:work_is_about_concept_eurovoc ?conceptEurovoc.
?conceptEurovoc skos:prefLabel "VAT rate"@en .
} LIMIT 1000
```

API to content download

This API will allow you to download the digital file. For example, from a browser, the URL http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a1.0006.01/DOC_1 will allow you to download the corresponding digital file.

Further Exploration

If you only have the **work URL**, you can use content negotiation to specify the desired format and language. For example, you can use content negotiation, in requesting an English PDF version of a document by using HTTP headers. For example, with the following URL and following headers:

```
http://publications.europa.eu/resource/cellar/b84f49cd-750f-11e3-8e20-01aa75ed71a15
Accept: application/pdf;type=pdfa1a
Accept-Language: en
```

or

```
http://publications.europa.eu/resource/oj/JOL_2014_001_R_0001_01
Accept: application/pdf;type=pdfa1a
Accept-Language: en
```

This can be possible using CURL, the application postman or a specif plugin on your browser.

In this method, the client sends an HTTP request with these headers, and the server returns the file in the requested format and language. This can be useful when building datasets in specific formats or for multilingual analysis.

Python Code for Automated Downloads

Here is a Python code snippet that uses the SPARQLWrapper library to automate the retrieval of documents from the repository.

1. It searches/lists the URL of digital content files according to the criteria in SPARQL queries, and
2. downloads the digital content file using the API and the python requests library.

```
from SPARQLWrapper import SPARQLWrapper, JSON
import requests

# sparql queries to knowledge graph endpoint to search/list content
# files

sparql =
SPARQLWrapper("https://publications.europa.eu/webapi/rdf/sparql")
sparql.setQuery("""
prefix cdm: <http://publications.europa.eu/ontology/cdm#>
prefix purl: <http://purl.org/dc/elements/1.1/>
select distinct ?item
where {
    ?work cdm:date_creation_legacy ?date.
    ?w a ?class.
    ?expr cdm:expression_belongs_to_work ?work ;
          cdm:expression_uses_language ?lang .
    ?lang purl:identifier ?langCode .
    ?manif cdm:manifestation_manifests_expression ?expr;
           cdm:manifestation_type "pdfa1a".
    ?item cdm:item_belongs_to_manifestation ?manif.
    FILTER (
        ?date > "2016-05-23T10:20:13+05:30"^^xsd:dateTime AND
        ?date < "2020-05-23T10:20:13+05:30"^^xsd:dateTime
    ).
    FILTER(
        ?class in (
            <http://publications.europa.eu/ontology/cdm#document_cjeu>,
            <http://publications.europa.eu/ontology/cdm#case-law>,
            <http://publications.europa.eu/ontology/cdm#summary_case-
law>,
            <http://publications.europa.eu/ontology/cdm#summary_case-
law_jure>
        )
    )
} LIMIT 1000
""")
sparql.setReturnFormat(JSON)
results = sparql.query().convert()

# Call to the  API to download the digital file

for result in results["results"]["bindings"]:
    doc_url = result["item"]["value"]
    response = requests.get(doc_url)
    file_name = doc_url.replace("/", ".") + ".pdf"
    with open(file_name, 'w', encoding='utf-8') as file:
        file.write(response.text)
    print(f"Downloaded: {file_name}")
```