Exposé

Research Project: Extending LinkedPapersWithCode (LPWC) Repositories in a more fine-grained Knowledge Graph

Abdul Rafay

abdul.rafay@mailbox.tu-dresden.de

5201784

Introduction

In the realm of software engineering and scientific research, effectively linking papers with associated code repositories is essential for enhancing reproducibility and accelerating knowledge transfer. The LinkedPapersWithCode (LPWC) initiative has made significant strides in bridging these domains by providing a platform that connects research literature with relevant code implementations. However, current systems often lack the granularity needed to capture the intricate relationships between different components of code, datasets, and research contributions. This research project aims to extend the LPWC repositories by developing a more fine-grained knowledge graph that will clearly delineate these relationships and enhance discoverability.

Objectives

- To construct a more detailed knowledge graph that comprehensively represents the relationships between research papers, code repositories, datasets, contributors, and related workflows.
- To identify and define key entities and their relationships in a way that reflects the multifaceted nature of software development and research contributions.
- To develop a framework for integrating and visualizing these relationships, facilitating easier access and understanding for researchers and developers alike.
- To enhance the interoperability and reusability of code repositories through better documentation of their connections to research outputs.

Methodology

The methodology for this research project involves a systematic approach to constructing a knowledge graph from GitHub repositories linked to the LinkedPapersWithCode (LPWC). The following phases outline the steps to be taken in the research process:

Data Extraction:

The initial phase involves extracting data from the GitHub repositories using the GitHub API. This will include metadata such as repository titles, descriptions, contributors, programming languages used, and relevant code files etc.

Data Analysis and Preprocessing:

Once the data is extracted, it will undergo a thorough analytical process. This includes cleaning the data to eliminate inconsistencies, handling missing values, and transforming data formats as necessary. Preprocessing steps may involve normalizing text fields, deduplication entries, and structuring the data for efficient modeling.

Modeling the Data as RDF Triples:

After preprocessing, the identified variables will be modeled into Resource Description Framework (RDF) triples. This modeling will capture the relationships and attributes of the entities involved in a format suitable for inclusion in a knowledge graph. Each triple will consist of a subject, predicate, and object to formalize the interconnections between different entities.

Constructing the Knowledge Graph:

With a substantial number of RDF triples, the knowledge graph will be constructed. The graph will then serve as a structured representation of the relationships among the various entities, facilitating enhanced data retrieval and analysis.

Data Retrieval Techniques:

Upon establishing a significant and coherent knowledge graph, various search techniques will be implemented to enable effective data retrieval. The following approaches will be employed:

RAG Technique:

Vector embedding's of the knowledge graph will be calculated to transform graph data into a vector space representation. This representation will allow the implementation of Retrieval-Augmented

Generation (RAG) techniques, which leverage external knowledge for improved search accuracy and relevance.

Fine-Tuning a Large Language Model (LLM):

A large language model will be fine-tuned utilizing the constructed knowledge graph as the corpus. This will enhance the model's ability to understand and generate contextually relevant responses based on the relationships and entities represented in the knowledge graph.

Classes

* Repository

- Properties:

title (String)

author (Person) → Relationship: Authored By

description (String)

abstract (String)

stars (Integer)

forks (Integer)

languages (List of Strings)

tags (List of Strings)

issues (List of Issues) → Relationship: Has

contributors (List of Persons) → Relationship: Includes

repositoryUrl (URL)

creationDate (Date)

sourceDirectories (List of Directories) → Relationship: Contains

* Person
    - Properties:
        name (String)
        mbox (Email)

* Issue
    - Properties:
        title (String)
        description (String)
        createdDate (Date)
        status (String)

* Directory
    - Properties:
        name (String)
        files (List of Files) → Relationship: Contains

* File
    - Properties:
        name (String)

fileType (String) (e.g., "Python", "JavaScript", "HTML")

size (Integer) (File size in bytes)

fileUrl (URL)

dependencies (List of Libraries) → Relationship: Requires

* Library

- Properties:

name (String)

version (String) (e.g., "1.0.0")

Relationships

Repository → Authored By → Person: Each repository is authored by one person (the author).

Repository → Has → Issues: Each repository can have multiple issues.

Repository → Includes → Person: Each repository can include multiple contributors.

Repository → Contains → Directories: Each repository contains multiple directories.

Directory → Contains → Files: Each directory can contain multiple files.

File → Requires → Libraries: Each file may have multiple libraries or dependencies that it requires.

Person can be the maker of, or a contributor to, multiple Repositories.

Example of Textual Representation of Relationships

[Repository] --(Authored By)--> [Person]

[Repository] --(Has)--> [Issue]

[Repository] --(Includes)--> [Person]

[Repository] --(Contains)--> [Directory]

[Directory] --(Contains)--> [File]

[File] --(Requires)--> [Library]

@prefix ex: <http://example.org/> .

ex:ResearchPaper1 ex:hasTitle "Deep Learning for Natural Language Processing" .

ex:ResearchPaper1 ex:hasAbstract "This paper explores deep learning techniques for various natural language processing tasks, including sentiment analysis, translation, and summarization." .

ex:ResearchPaper1 ex:hasSummary "We propose a novel architecture that improves sentiment analysis results over traditional methods." .

ex:ResearchPaper1 ex:containsSentence ex:Sentence1 .

ex:Sentence1 ex:hasContent "The model consistently outperforms baseline methods in the sentiment analysis task." .

ex:Sentence1 ex:cites ex:ResearchPaper2 .

ex:Sentence1 ex:cites ex:ResearchPaper3 .

ex:ResearchPaper2 ex:hasTitle "Advancements in Neural Networks" .

ex:ResearchPaper2 ex:hasAbstract "This paper presents new architectures that enhance the performance of neural networks, focusing on convolutional and recurrent networks." .

ex:ResearchPaper2 ex:hasSummary "We discuss the impact of these advancements on tasks like image recognition and language modeling." .

ex:ResearchPaper2 ex:containsSentence ex:Sentence2 .

ex:Sentence2 ex:hasContent "Convolutional neural networks (CNNs) have shown remarkable performance on image classification." .

ex:Sentence2 ex:cites ex:ResearchPaper3 .

ex:ResearchPaper3 ex:hasTitle "Natural Language Understanding with Neural Networks" .

ex:ResearchPaper3 ex:hasAbstract "This paper reviews the application of neural networks in natural language understanding, highlighting their effectiveness in parsing and named entity recognition." .

ex:ResearchPaper3 ex:hasSummary "We analyze different neural network architectures and their contributions to the field." .

ex:ResearchPaper3 ex:containsSentence ex:Sentence3 .

ex:Sentence3 ex:hasContent "Recent models achieve state-of-the-art results in named entity recognition tasks." .

ex:Sentence3 ex:cites ex:ResearchPaper1 .

ex:Sentence3 ex:cites ex:ResearchPaper2 .

ex:ResearchPaper1 ex:cites ex:ResearchPaper3 .

ex:ResearchPaper2 ex:cites ex:ResearchPaper1 .

ex:ResearchPaper2 ex:cites ex:ResearchPaper3 .

https://issemantic.net/rdf-visualizer

https://ontopea.com/