Exploring ML solutions for Determining Python Software Repository Similarity

Supervisor: Rosa Filgueira, Lei Fang

Student: Leyu Zhang (220016784), Honglin Zhang (220032952)

1. Description

Code-to-code similarity refers to the semantics similarity between two code snippets, which is useful to detect cloned code and find the similar code snippets and similar applications. This project aims to explore machine learning techniques for calculating code similarity and extend it to repository similarity by metadata analysis [1] and latent Dirichlet allocation [2].

Firstly, we will leverage them to convert code into numerical representations and measure the semantic similarity between code snippets. By training these models on Python code datasets, we will evaluate their performance in identifying cloned code and finding similar code snippets.

Then we will try to extend from comparing code similarity to repository similarity by combining it with metadata, document, and source code [1]. we will integrate the inspect4py [3] framework, which extracts relevant information from software repositories such as functions, classes, methods, documentation, dependencies and so on. In this part, we would understand and replicate some techniques based on embedding approaches [4] such as Doc2Vec which can transform the documents into vectors; and Code2Vec which can transform the source code into vectors, which are defined in Repo2vec paper [1].

Finally, we will try to merge different embedding approaches including metadata2vec and others into one model and compare it with LDA model. If it is possible, the LDA and embedding approaches will be merged into one model and the performance of the model will be evaluated. The paper Repo2Vec [1] mentions the way to compare the Java code similarity but the project will replicate the work of Repo2vec for Python Code repositories.

2. Objectives

There are several interim goals in the project. By the way, A and B represents different teammates.

2.1 Primary objectives

1. Understand and replicate embedding approaches for Python code similarity.

1) Metadata2Vec (A) ---- replicating the work done in Repo2Vec paper using metadata for creating an embedding to represent a repository using metadata information.
2) Doc2Vec (A+B) ---- replicating the work done in Repo2Vec paper using docstrings for creating an embedding to represent a repository using docstring information.
3) Code2Vec (A+B) ---- replicating the work done in Repo2Vec paper using codes for creating an embedding to represent a repository using code information.
4) Repo2Vec (A+B) ---- merging the three previous embedding techniques (Meta2vec, Doc2vec, Code2vec) into a single embedding, repo2vec, to represent a repository. This also replicates the work done in Repo2vec paper.

Metadata includes data content, structure, and format. Metadata2Vec will transform meta data into vectors.

2. Understand and replicate machine learning models for Python code similarity.
   1) Transformer [5] (A) for creating embeddings at docstring, metadata and code levels. Note that maybe different transformers might need to be applied for generating each type of embedding. We will start studying the performance of UnixCoder Transformer [6] for generating these embeddings.
   2) LDA [2] (B) calculate the frequency of the words in the repositories and build a document-frequency matrix. The matrix will be input into LDA model, and a topic representation is fitted. According to different parameters of LDA model, the different types of topics can be represented.
3. Using LDA to extract the topic of a repository. (B)
4. Develop Python scripts based on Embedding approaches and LDA to evaluate the similarity between some Python codes. (A+B)
5. Compare the performance of two methods. (A+B)

2.2 Secondary objectives:

1. Train models and evaluate them on a public database. (B)
2. Merge metadata2vec and doc2vec into one model. (A)
3. Merge embedding approaches and LDA into one model. (A+B)

3. Ethics

This project has no ethical considerations as outlined in the assessment form.

4. Resources

CS lab machines and Collab Notebooks.

5. Data Set

Two public data set will be used for evaluating our methods developed within the project:

1. https://paperswithcode.com/

2. https://github.com/vinta/awesome-python

6. Schedule

Weekly meeting with the supervisors on Friday.

7. Reference

[1] Repo2Vec: A Comprehensive Embedding Approach for Determining Repository Similarity https://arxiv.org/pdf/2107.05112.pdf

[2] Example: Amortized Latent Dirichlet Allocation https://pyro.ai/examples/lda.html

[3] Inspect4py: A Knowledge Extraction Framework for Python Code Repositories https://github.com/SoftwareUnderstanding/inspect4py

[4] The Ultimate Guide To Different Word Embedding Techniques In NLP https://www.kdnuggets.com/2021/11/guide-word-embedding-techniques-nlp.html

[5] How Transformers Work https://towardsdatascience.com/transformers-141e32e69591

[6] UnixCoder: https://arxiv.org/abs/2203.03850