Context-Aware Notebook Search in a Jupyter-Based Virtual Research Environment

Na Li*, Siamak Farshidi*, Riccardo Bianchi†, Spiros Koulouzis†, Zhiming Zhao*†
*Multiscale Networked System, Informatics Institute, University of Amsterdam, Netherlands,
†LifeWatch ERIC Virtual Lab and Innovation Center (VLIC), Amsterdam, Netherlands
{n.li | s.farshidi | r.bianchi | s.koulouzis | z.zhao}@uva.nl

***Abstract*—Computational notebook environments such as the Jupyter play an increasingly important role in data-centric research for prototyping computational experiments, documenting code implementations, and sharing scientific results. Effectively discovering and reusing notebooks available on the web can reduce repetitive work and facilitate scientific innovations. However, general-purpose web search engines (e.g., Google Search) do not explicitly index the contents of notebooks, and notebook repositories (e.g., Kaggle and GitHub) require users to create domain-specific queries based on the metadata in the notebook catalogs, which fail to capture the working contexts in the notebook environment. This poster presents a Context-aware Notebook Search Framework (CANSF) to enable a researcher to seamlessly discover external notebooks based on semantic contexts of the literate programming activities in the Jupyter environment.**

***Index Terms*—computational notebook, Jupyter notebook, notebook search, data science, code reuse**

I. Introduction

A computational notebook, e.g., Jupyter notebook, provides a literate programming [1] environment for researchers to prototype and execute computational experiments. It can effectively support exploratory and iterative computational processes, and the documented notebooks can be easily shared for scholarly communication and code reuse [2].

During past years, more than a billion notebooks have been created[1], which become an important type of web resources for researchers to check related work, reuse existing implementations, and expedite scientific innovations. However, general search engines mainly focus on HTML documents and multimedia and do not explicitly index notebooks. Moreover, existing search tools rely on queries explicitly generated by users and fail to capture the programming contexts in the notebook environment.

Several barriers hamper the finding, accessing, and utilizing (namely FAIRness [3]) of notebooks. First, notebooks available on the web have a diverse quality of documentation and source codes. Most notebooks published on GitHub are "not narratives but collections of scripts with loose notes" [4], which makes the execution difficult and thus impairs the reproducibility of the experiments contained in these notebooks. Second, only a small portion of notebooks created by scientific communities have been curated and collected by public notebook repositories or catalogs. Many research infrastructures, e.g., ENVRI [5], still focus on the assets of research data and web services in their catalogs; less attention has been paid to notebooks. Yet most catalogs treat the notebook as a single digital object or part of a project and do not provide searching over notebook contents.

This paper aims to tackle notebook search challenges in the context of the research activities conducted in a Jupyter environment. We propose a Context-aware Notebook Search Framework (CANSF), a semantic notebook search system that can derive context aware queries from users' working notebooks and search for related notebooks. We prototype the proposed system and integrate it with the Jupyter environment.

II. Problem statement and related work

Notebooks group content into chunks called *cells*. There are two basic types of cells: *code cells* containing code fragments, and *Markdown cells* for narrative descriptions. Notebooks can be searched based on metadata, code in code cells, or textual descriptions in Markdown cells. Research on notebook search is still in its infancy, with little prior work. NBSearch [6] supports semantic code search in an extensive notebook collection and visual result exploration. It treats code cells as search units and does not consider Markdown cells. JupySim [7] is a content-based notebook search system that takes codes, tabular data and libraries as queries. However, these systems work as standalone search tools, do not have seamless integration with the Jupyter environment, and most of all, do not consider contextual information.

III. Context-Aware Notebook Search Framework

To address the challenges we highlighted we propose the Context-aware Notebook Search Framework (CANSF). The main focuses of the system are: 1) context-awareness during searching and 2) support for semantic notebook representation. Users that extend their Jupyter environments with CANSF achieve increased efficiency in their notebook programming.

A. System Design

Fig. 1 illustrates the high-level architecture of the framework. It consists two parts: a search agent to be installed on the client side, e.g., a Jupyter environment, and a search backend.

The *Search agent* is integrated in the notebook environment and observes the activities conducted by the user. By analyzing

[1] https://blog.jetbrains.com/datalore/2020/12/17/we-downloaded-10-000-000-jupyter-notebooks-from-github-this-is-what-we-learned/

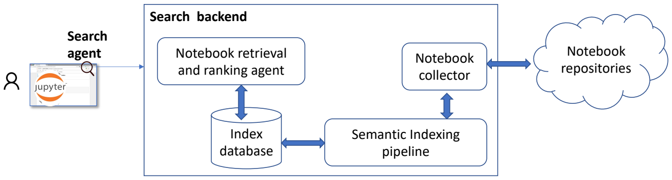


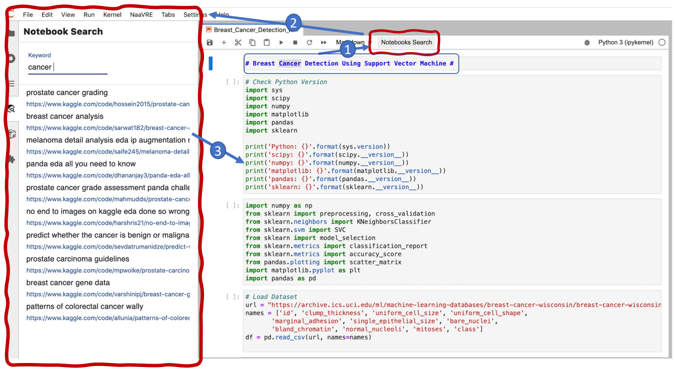
Fig. 1. The high-level structure of proposed Context-aware Notebook Search Framework (CANSF).

the cell content that the user is working on, the search agent captures the contextual information, e.g., types of scientific problem, workflow components, AI models and data sets from the cells, and creates context aware queries to the backend.

The backend consists of four components: 1) a *Notebook collector* crawls notebooks from different web locations (e.g., GitHub and Kaggle); 2) a *Semantic indexing pipeline* digests raw notebooks and creates indexes by using keywords, topic modelling and knowledge graphs; 3) an *Index database* stores the indexes and make them available for searching; 4) a *Notebook retrieval and ranking agent* processes the queries sent by the *search agent*, and retrieves matched notebooks from the *Index database* based on their similarities.

B. Implementation

The current software is implemented in the context of several EU projects i.e., H2020 ENVRI-FAIR, CLARIFY, and BlueCloud. We integrate our notebook search framework with the Jupyter notebook environment as an extension tool to enable in-site notebook search for supporting the development of data analytic pipelines. Fig. 2 shows the user interface for notebook search in the Jupyter notebook environment.



cancer

prostate cancer grading
https://www.kaggle.com/code/hossein2015/prostate-can
breast cancer analysis
https://www.kaggle.com/code/sarwat182/breast-cancer-
melanoma detail analysis eda ip augmentation
https://www.kaggle.com/code/saife245/melanoma-detail
panda eda all you need to know
https://www.kaggle.com/code/dhananjay3/panda-eda-all
prostate cancer grade assessment panda challe
https://www.kaggle.com/code/mahmudds/prostate-canc
no end to images on kaggle eda done so wrong
https://www.kaggle.com/code/harshris21/no-end-to-imag
predict whether the cancer is benign or maligna
https://www.kaggle.com/code/sevdatrumanidze/predict-
prostate carcinoma guidelines
https://www.kaggle.com/code/mpwolke/prostate-carcino
breast cancer gene data
https://www.kaggle.com/code/varshinipj/breast-cancer-g
patterns of colorectal cancer wally
https://www.kaggle.com/code/allunia/patterns-of-colore

```
# Breast Cancer Detection Using Support Vector Machine #
```

```
[ ]: # Check Python Version
     import sys
     import scipy
     import numpy
     import matplotlib
     import pandas
     import sklearn

     print('Python: {}'.format(sys.version))
     print('scipy: {}'.format(scipy.__version__))
     print('numpy: {}'.format(numpy.__version__))
     print('matplotlib: {}'.format(matplotlib.__version__))
     print('pandas: {}'.format(pandas.__version__))
     print('sklearn: {}'.format(sklearn.__version__))
```

```
[ ]: import numpy as np
     from sklearn import preprocessing, cross_validation
     from sklearn.neighbors import KNeighborsClassifier
     from sklearn.svm import SVC
     from sklearn import model_selection
     from sklearn.metrics import classification_report
     from sklearn.metrics import accuracy_score
     from pandas.plotting import scatter_matrix
     import matplotlib.pyplot as plt
     import pandas as pd
```

```
[ ]: # Load Dataset
     url = "https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin
     names = ['id', 'clump_thickness', 'uniform_cell_size', 'uniform_cell_shape',
            'marginal_adhesion', 'single_epithelial_size', 'bare_nuclei',
            'bland_chromatin', 'normal_nucleoli', 'mitoses', 'class']
     df = pd.read_csv(url, names=names)
```

Fig. 2. User interface of CANSF in a Jupyter environment.

When a user is developing an experiment, the search agent can analyze the markdown cell the user is working on, derive the keywords for searching related notebooks (step 1), and search the relevant notebook by calling the search backend. The search results will be presented as a panel on the left side in the Jupyter environment (step 2). The user can select relevant ones, obtain the content (code or notebooks), and further edit them in the working experiment (step 3). These steps are triggered by the user explicitly when the user needs to look for related work to the working notebook.

Solid and rich indexes are essential to support context matches of the retrieved notebooks with the user operational status. To measure this we tested the performance of our semantic indexing pipeline. We collected 37 queries from 7 researchers from the CLARIFY project. In this case the contextual information is primarily about medical imaging and related AI models. At the same time we collected and indexed 1250 notebooks crawled from the Kaggle platform[2]. Our experiments show that CANSF can provide the users with highly relevant external notebooks that well match their query and context.

IV. Conclusion and Future Work

In this poster, we present CANSF, Context-aware Notebook Search Framework, that increases the discover-ability of computational notebooks. In the future, we will improve notebook search performance by introducing computing-related factors into notebook ranking, e.g., execution time. Furthermore, given that notebooks can be seen as a connection for different research objects, we will investigate a combined search paradigm, e.g., searching for datasets and machine learning models simultaneously. Finally, it is our goal to include CANSF into the general architecture of the NaaVRE - Notebook-as-a-Virtual-Research-Environment [8].

Acknowledgment

This work has been partially funded by the European Union's Horizon 2020 research and innovation program by the project CLARIFY under the Marie Skłodowska-Curie grant agreement No 860627, by the ARTICONF project grant agreement No 825134, by the ENVRI-FAIR project grant agreement No 824068, by the BLUECLOUD project grant agreement No 862409, by the LifeWatch ERIC.

References

[1] D. E. Knuth, "Literate programming," *The computer journal*, vol. 27, no. 2, pp. 97–111, 1984.

[2] J. M. Perkel, "Why jupyter is data scientists' computational notebook of choice," *Nature*, vol. 563, no. 7732, pp. 145–147, 2018.

[3] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne *et al.*, "The fair guiding principles for scientific data management and stewardship," *Scientific data*, vol. 3, no. 1, pp. 1–9, 2016.

[4] A. Rule, A. Tabard, and J. D. Hollan, "Exploration and explanation in computational notebooks," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018, pp. 1–12.

[5] A. Petzold, A. Asmi, A. Vermeulen, G. Pappalardo, D. Bailo, D. Schaap, H. M. Glaves, U. Bundke, and Z. Zhao, "ENVRI-FAIR - Interoperable Environmental FAIR Data and Services for Society, Innovation and Research," in *2019 15th International Conference on eScience (eScience)*. San Diego, CA, USA: IEEE, Sep. 2019, pp. 277–280. [Online]. Available: https://ieeexplore.ieee.org/document/9041704/

[6] X. Li, Y. Wang, H. Wang, Y. Wang, and J. Zhao, "Nbsearch: Semantic search and visual exploration of computational notebooks," in *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, 2021, pp. 1–14.

[7] M. Horiuchi, Y. Sasaki, C. Xiao, and M. Onizuka, "Jupysim: Jupyter notebook similarity search system," *Open Proceedings*, 2022.

[8] Z. Zhao, S. Koulouzis, R. Bianchi, S. Farshidi, Z. Shi, R. Xin, Y. Wang, N. Li, Y. Shi, J. Timmermans *et al.*, "Notebook-as-a-vre (naavre): from private notebooks to a collaborative cloud virtual research environment," *arXiv preprint arXiv:2111.12785*, 2021.

[2] https://www.kaggle.com/