Fall 2023
FORWARD DATA LAB

Research Report on Science Analytics

Reading Papers, Fetching Faculty Information and Doing Analysis

Haowen Zheng

1 Introduction

During this summer research, I have read lots of papers related to science analytics and participated in group meetings to report my weekly progress. The primary focus of the study is: how to use known faculty names and affiliations to accurately find faculty in databases and obtain their publications, image link, and other information. To achieve this, I explored various databases such as openalex, dblp, semantic scholar, and google scholar, and employed multiple methods to handle names and affiliations. In the following sections, I will provide a detailed explanation of the work I undertook during this research.

2 Fetching Author Publications with Name and Affiliation

We are provided with a faculty csv file containing CS department faculty names and affiliations. The goal is to fetch their publications and other information, which can be used to build an academic graph in Neo4j. Then an important question is that which academic database is a better source.

The database needs to provide api allowing finding author by both name and affiliation. I mainly check for 3 databases: OpenAlex, Semantic Scholar and Google Scholar. Among them, semantic scholar often do not contain affiliation field because an author needs to register at their website to add affiliation. Thus, I choose OpenAlex and Google Scholar to do the comparison.

2.1 First Step: Name Cleaning

The first step is to clean the name. Some names have "MS" or "PhD" so we need to remove them. eg. "Ye Xia PhD" → "Ye Xia". Also, some names contain unicode, so we need to convert them before searching. eg. "\u00d6mer E\u011fecio\u011flu" → "Ömer Eğecioğlu".

Another part is solving name aliasing. The same person may have different names in different databases. For example, "James Reginald Miller" has name aliases like "James R. Miller", "James Miller", "J. R. Miller", "J. Miller", "J. Reginald Miller", "Reginald Miller". I create a list of name aliases like this and search each of them in the database to enhance the probability of finding the author.

2.2 OpenAlex

1. **Approach:** OpenAlex provides an api allowing searching author by various properties. The api is: `https://api.openalex.org/authors?filter=display_name.search:{}, last_known_institution.country_code:US`. The parameter is the author name and the country code is used to filter out non-US authors. This api returns a json file containing all the authors that match the name. Then I use a python library fuzzywuzzy to calculate the similarity between the affiliation I search and the affiliations in the json result. If the similarity is greater than 80, I consider it as a match. Finally, the author's publications can be easily found by `https://api.openalex.org/works?filter=author.id:{}`.

2. **Result and Issue:** Among all 5596 faculties, I can find 4239 authors with matched name and affiliation in OpenAlex. There are still around 1k authors not found and the reasons might be that the author is not exist in OpenAlex or the last known institution field in OpenAlex is outdated. Besides, there is one more issue that among the 4239 faculties found, nearly three fourths of them have more than one records in OpenAlex. It means that even if the name and affiliation are matched, we still get multiple results.(some may have more than 100 records in the database) I manually check some records with same name and affiliation, and find that some results are the same person in different periods and OpenAlex does not merge them together. Fig1 shows an example of this issue. As

a result, after searching name and affiliation using OpenAlex api, we still need to merge those records or simply choose one with the most citation count.

Figure 1: Same Person With Multiple Records

2.3 Google Scholar

Another great academic source is google scholar. It does not have multiple records for the same author and the publications of one author can be found at the author profile page. However, it does not provide an official api. Thus, the only way to fetch author profiles is web scraping. There are several existing ways to do web scraping. The first one is through Serpapi. (`https://serpapi.com/google-scholar-api`) It is a paid api that allows user to customize search query, pagination option or some advanced filter options. The second one is a github project: scrape-google-scholar-py.(`https://github.com/dimitryzub/scrape-google-scholar-py`) It would simply scrape the searching page and return a classified json result. However, after I tried it to process more than 200 authors, it's blocked by google scholar. Maybe I need to use some customized proxies. The third one is the scholarly python library. (`https://scholarly.readthedocs.io/en/stable/quickstart.html`) It's able to solve the CAPTCHA issue to prevent blocking from google scholar, but it still has a everyday using limit. I use scholarly as the final method.

1. **Approach:** Since google scholar supports fuzzy search, for example, you can still get Alman Josh's profile if you search Josh Alman or Alman J. Thus the name aliasing process can be simplified. For John Smith K, I would search for John Smith K, Jhon Smith, Jhon and Smith. Then my search query is name + affiliation. I noticed that google scholar also has email field, so if the name + affiliation pattern does not work, I would try searching name + email.

2. **Result and Issue:** I searched a total 500 CS faculties in google scholar and get 459 results. Obviously, it has a lower missing rate than OpenAlex. And google scholar ensures that there is no multiple results for same author and affiliation. However, there are still about 8.2% authors not found. One possible reason is that Google Scholar allows authors to define their own affiliation names, resulting in lack of consistency.

2.4 Combined Result

Then I consider to combine the searching result for both OpenAlex and Google Scholar. I use Google Scholar as a primary searching source. If the author is not found in Google Scholar, I would try to search him/her in OpenAlex. After doing this, among 500 CS faculties, there are only 26 authors not found. I manually check their names and find that most of them are not

exist in both datasources. Finally, I append author's Google Scholar id or OpenAlex id to the csv file.

3 Reformat OpenAlex Object and Create Neo4j Nodes

After getting the accurate faculty data, we can import them into graph database Neo4j to analyze the relationship between elements. Here's what I did at the beginning of the research: reformatting OpenAlex object and loading them into Neo4j.

OpenAlex contains different types of nodes including: work, author, concept, funder, institution, publication and source. And these types are linked with certain properties. For example, work nodes are connected to author nodes with the "authorship" property in work node. In order to import these nodes into Neo4j, we need to reformat the OpenAlex object and create nodes with the corresponding properties. The original OpenAlex object is a multiple-level json file. Then I extract the important properties and flatten the object into single-level python dictionary, which is suitable for transforming to csv and loading into Neo4j.

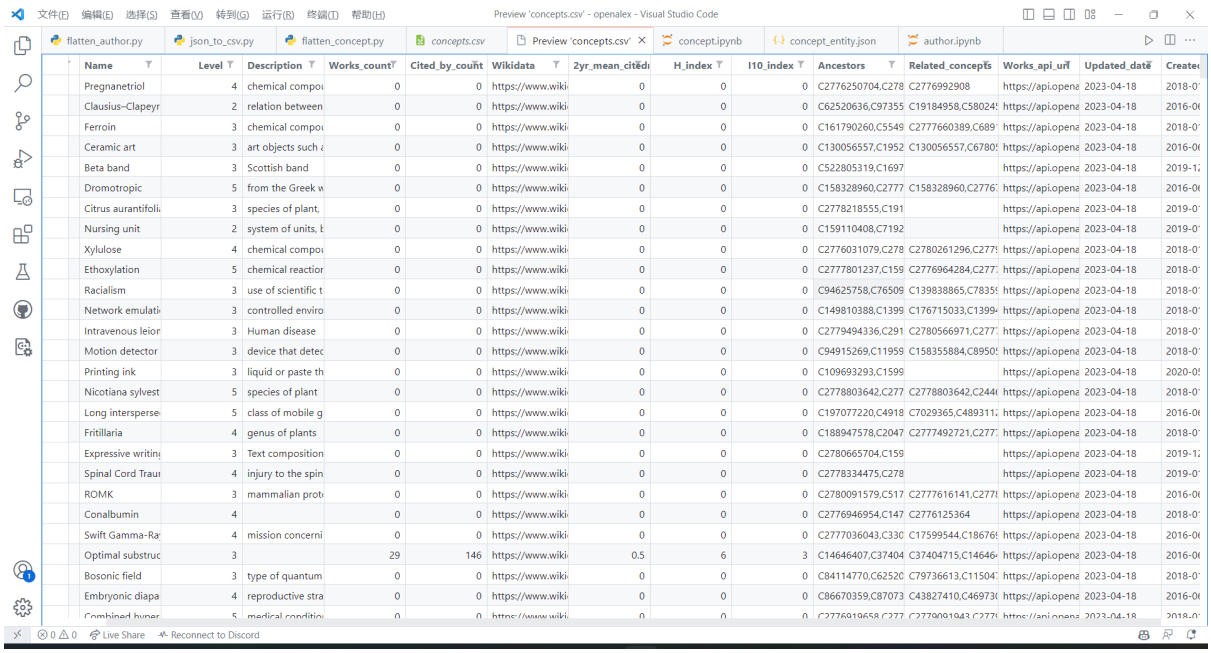

Figure 2: Concept Nodes After Reformatting

After getting the reformatted csv, we needs to create nodes in Neo4j using LOAD CSV tool.(`https://neo4j.com/developer/guide-import-csv/`) I classify the properties into self-property and relationship-property. Self-property is the property of the node itself, for example, the name of the author. Relationship-property is the property of the relationship between two nodes, for example, the authorship property between author and work. Then I create nodes with self-property and create relationships with relationship-property. The following figures show the final graph result.

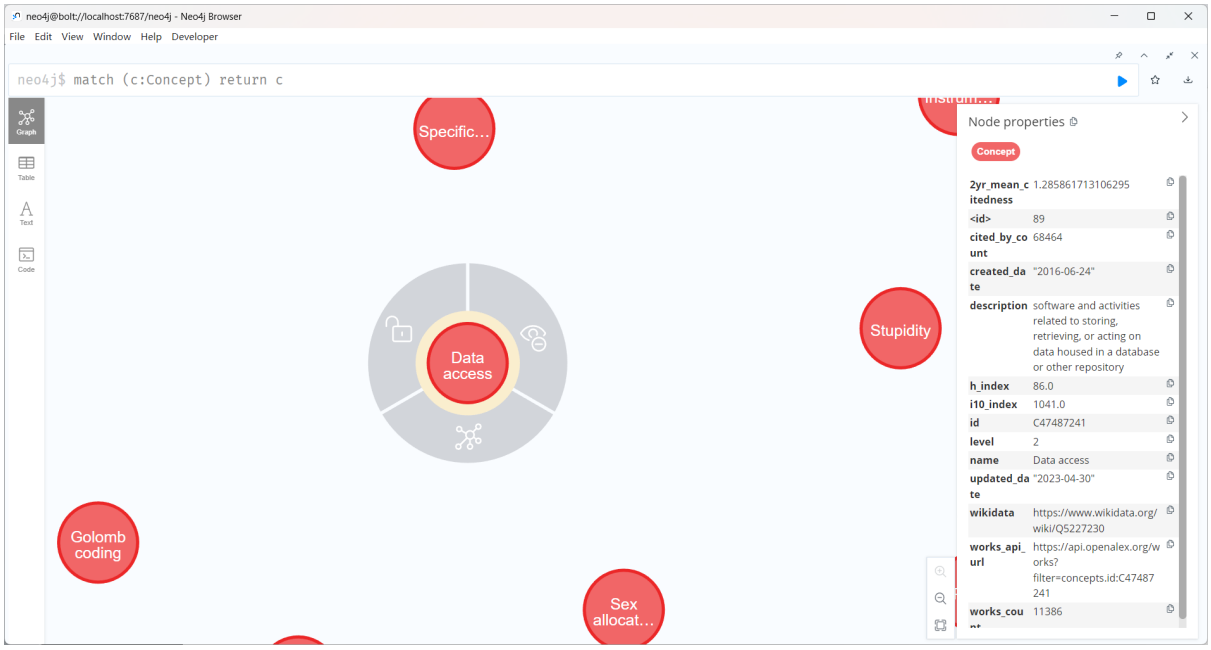

Figure 3: Concept Node Example in Neo4j

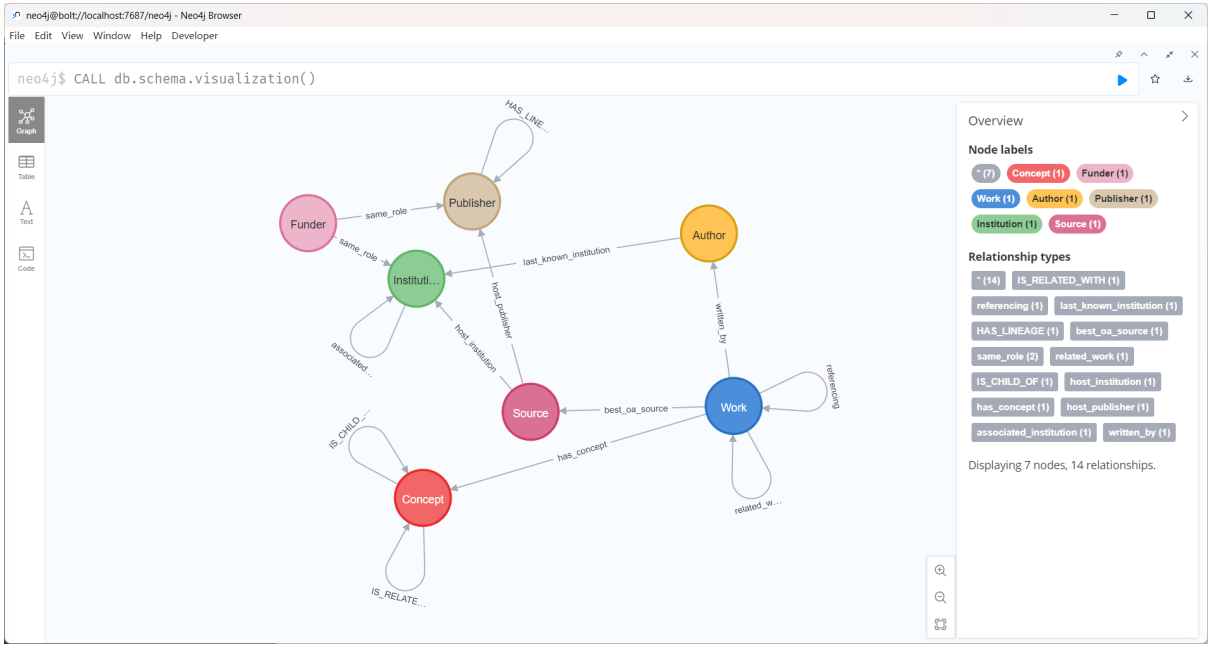

Figure 4: Schema of the Graph

4 Face Recognition

For future work, we can divide the faculty by demographic attributes like age, gender or race and analyze their publication rate. To specify these attributes, one way is through face recognition.

1. **Approach:** Here's two tools I found to do face recognition: OpenCV and FairFace. OpenCV is a popular computer vision library. (`https://docs.opencv.org/3.4/da/d60/tutorial_face_main.html`) It offers tools for face recognition using algorithms like Eigenfaces, Fisherfaces, and Local Binary Patterns Histograms. Faces in images are firstly detected using DNN methods. Then you can use their pretrained model to predict the age or gender of that person. FairFace is a dataset designed to address biases in face recognition and classification systems. (`https://github.com/joojs/fairface`) It offers a balanced representation across race, gender, and age, making it a valuable resource for training fair and equitable facial analysis models. Unlike traditional datasets, FairFace prioritizes diversity, enabling developers to create more inclusive facial recognition applications.

2. **Issue:** After our weekly meeting, I recognized a significant concern with face recognition. The profile image on Google Scholar doesn't always reflect the author's current appearance, making it unreliable for predicting demographic attributes.

5 Reflection: What I Have Learned During This Research

During this research, my understanding of science analytics has deepened substantially. Every week, I read related papers and discussed project progress in our meetings. I got to know some academic databases like Openalex and Google Scholar and face recognition tools like OpenCV or Fairface. Furthermore, I acquired knowledge about the Neo4j graph database and obtain practical experience in data processing.

One important thing I learned is how important accurate data is for any research project. We've discussed this issue and tried multiple methods during the process, which made me realize that doing research is a self-motivated task. When facing any troubles, we should always keep asking what caused this issue and how do we improve it. When reading papers, professor Kevin told us that we should always focus on how does the author come up with this idea and how does the author design the experiment. We are encouraged to think in the author's perspective and try to understand author's research process.

6 Future Work Suggestion

I think we're able to retrieve a relatively high quality data of faculty information using the methods we provided. Then we can classify these authors by certain attributes(age, gender, race, academic position) and see how these attributes affect their publication rate.

Additionally, we have the capability to construct academic graphs comprising nodes for authors, works, and institutions. Then we can analyze co-authorship across institutions. For instance, it might be observed that authors affiliated with prestigious institutions exhibit a stronger inclination to collaborate with peers from similarly esteemed institutions. Does it mean that there exists an inequality in academic collaboration based on institution prestige rank? I think these questions are insteresting to explore and we may find more topics to research if we complete the academic graph and analyze it more deeply.