UPenn:IEEE Practicum - Fall 2023

Building an Intelligent Web-scraping Model for Individual-level Scholarly Information

Sam Thudium, Abhinav Bandaru, Mikaela Spaventa,
Rut Vyas, Ruchika Singh

Penn Engineering
UNIVERSITY of PENNSYLVANIA

Outline

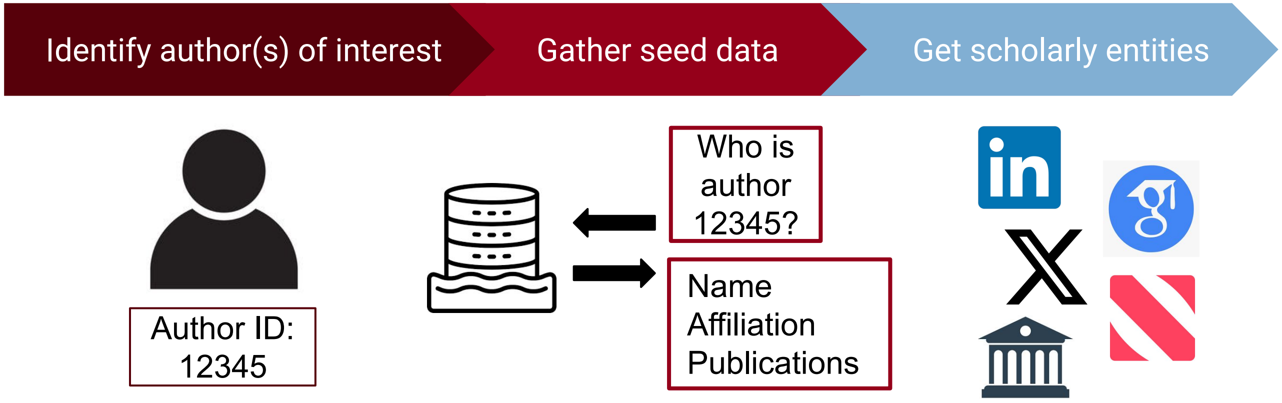
1. Problem definition
2. Approaches:
   - Top-Down (Common Crawl Data Repository)
   - Bottom-Up (Google Programmable Search Engine)
3. Final Product: `scholarscraper` package
   - Key Features
   - Results
   - Capabilities and Limitations
4. Further Discussion and Development

Problem Statement

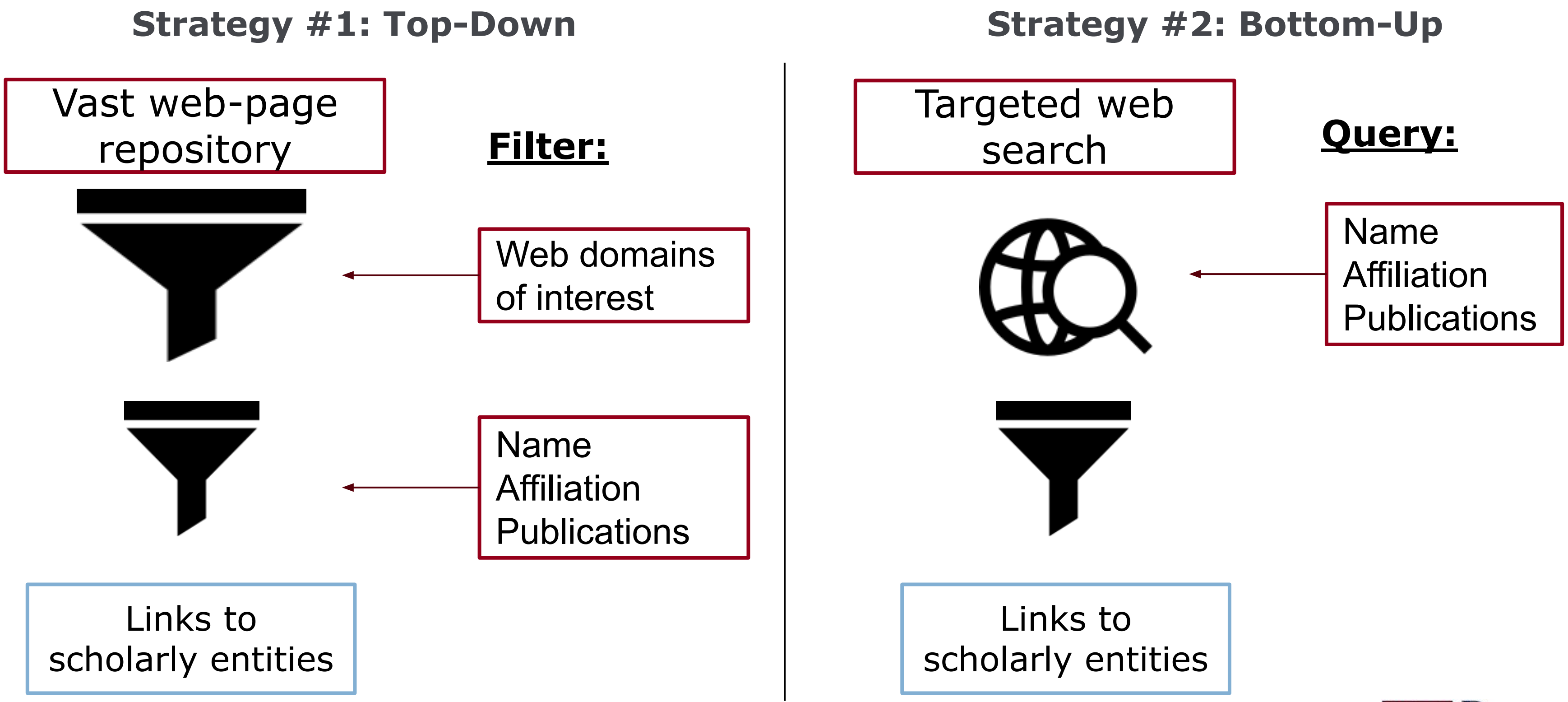
Project directive: create an intelligent web-scraping model capable of identifying links (active URLs) to "scholarly entities" related to authors in the IEEE data lake.

What are scholarly entities?: data sources not contained in the IEEE data lake that pertain directly to an author's professional work.

Data needed to begin (seeds): individual-level data queried from the data lake; name, primary affiliation, publication history.



Web-Scraping Approach Models



Top-Down Approach

Common Crawl

Scraping Model #1: Common Crawl ("Top-Down")

What is Common Crawl?

A free, open repository of web-pages scraped from across the internet[1].

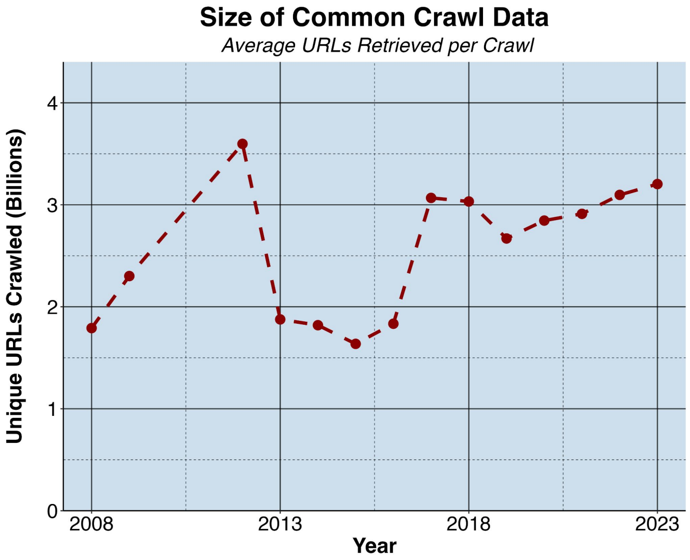
Data Size: Tens of petabytes with individual crawls amounting to 100-200 terabytes

Terms:

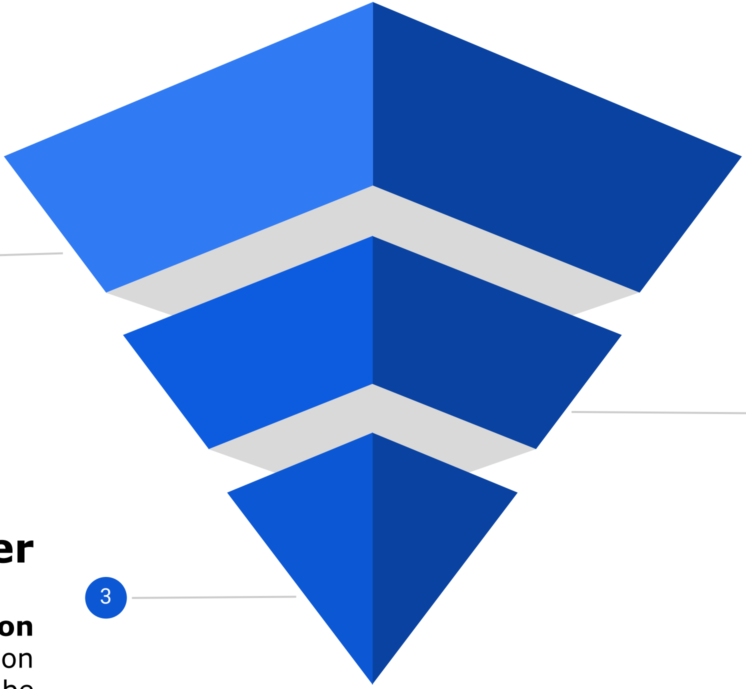
- Index: each new crawl characterized by the year and a crawl number
- WARC (Web ARChive Format): files which store the raw crawl data

Common Crawl has been used:

- By Facebook Research[2]
- As training data for numerous machine translation models[3]



Common Crawl - Process Overview



1

Indexer

Creates a **list of indexes** to be crawled based on the user input.

2

URL Filter

Filters based on the **domains** (github, twitter, wired) and requests **all** the urls from that domain in that index

3

WARC Filter

Uses **author seed information** (name, affiliation and publication keywords) to filter **WARC files** to be used for downstream processes

Common Crawl - Results Structure

User Input:

- Index: 2023-40
- Domain: Github
- Seed Info:

  fake_author = {123456789: {'authorname': 'Jesse Chen', 'affiliation': 'University ABC',
  'papertitle' : ['Important work Volume 1',
  'Applied Machine Learning Techniques’]}}

Output Format:

{authorid:
  {Domain:
    [(URL 1,WARC 1), (URL 2,WARC 2), ……]
  }
}

Eg:

{123456789:{'github.com/*':[(https://github.com/0xjessel, ‘Skip to content Toggle navigation)
…]}}

Penn Engineering
UNIVERSITY of PENNSYLVANIA

Common Crawl - Capabilities and Limitations

Capabilities:

- Create **search queries** for any author in the IEEE data lake
- Perform top-down style filtering on **petabytes** of data
  - Limit to high probability web domains
  - Search subset for author seeds
- Fast, **asynchronous search** scales to size of author set

Limitations:

- Free, open resource with **limited capacity** to handle requests in volume
- Respectful scraping behavior of CC; few or no links to important scholarly domains (**Scholar, LinkedIn, Twitter**)
- **Data size**; with our computational limits, one-time download and static querying infeasible

Bottom-Up Approach

Google API

Scraping Model #2: Google API (“Bottom-up”)

Objective: To utilize web scraping tools, such as BeautifulSoup, for extracting information from across the entire web.

Scope: Our focus was to gather a broad range of data from various web sources.

CHALLENGES:

Vastness of Data: Difficulty in efficiently extracting relevant data due to the immense volume of web information.

Relevance and Quality: Necessity to sift through a vast amount of data to find relevant information.

Scraping Model #2: Google API ("Bottom-up")

Initial Scraping Research:

- SERP API
- Scholarly python library

SerpApi

Pros of these options:

- Allow user to capture data from difficult sites (Google Scholar)

Flaws in these options:

- Proxy-based evasion of scraping limits is prohibited by Google

Final Choice:
Google Custom Search Engine

Product:

- Server-side access to a Google Search endpoint
- Allows curated search based on programmatically generated queries
- Not subject to risks of scraping at scale

Google API - Pipeline Overview

ID: 12345
Name
Affiliation
Publications

QUERYING → **SEARCH** → **CATEGORIZE** → **FILTERING**

Query author info from the IEEE data lake and produce **seed search strings** for the Google API

Author seeds used to search with the **Google Custom Search Engine**

Categorize URLs obtained from step 2 using **defined rules** into **buckets** like LinkedIn, Github, Google Scholar, etc.

The categorized links are further **filtered** to return only those **relevant to the author**[4,5]

Penn Engineering
University *of* Pennsylvania

Results Example: Google API - Pipeline in Action

Author: Michael Kearns
Affiliation: University of Pennsylvania
Top publications: "An Introduction to Computational Learning Theory",
"Near Optimal Reinforcement Learning in Polynomial Time"

Bucket/ Filter status						
Unfiltered	1	1	21	1	1	13
Filtered	1	1	2	1	1	10
Filtered and **limited (2)**	1	1	2	1	1	2

Google API - Pipeline in Action

Raw output JSON (Filtered/Limited):


```
"Michael Kearns": {
        "linkedin": [
            "https://www.linkedin.com/in/michael-kearns-0951337"],
        "github": [
            "https://github.com/mvcisback/lstar",
            "https://github.com/mcitoler/learning-theory"],
        "twitter": [
            "https://twitter.com/mkearnsupenn?lang=en"],
        "news": [],
        "scholar.google": [
            "https://scholar.google.com/citations?user=8iQk0DIAAAAJ&hl=en"],
        "dblp": [
            "https://dblp.org/pid/78/6858"],
        "edu": [
            "https://economics.sas.upenn.edu/people/michael-kearns",
            "https://www.cis.upenn.edu/~mkearns/"]
    }
```


Example web-pages:

Personal site:

LinkedIn profile:

Michael Kearns · 2nd

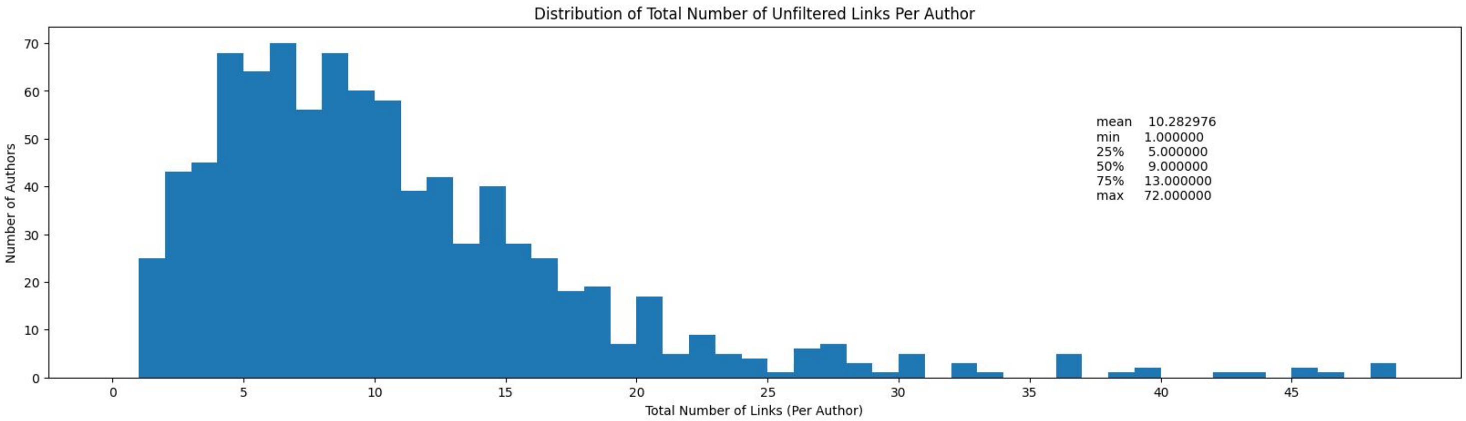
Professor at University of Pennsylvania

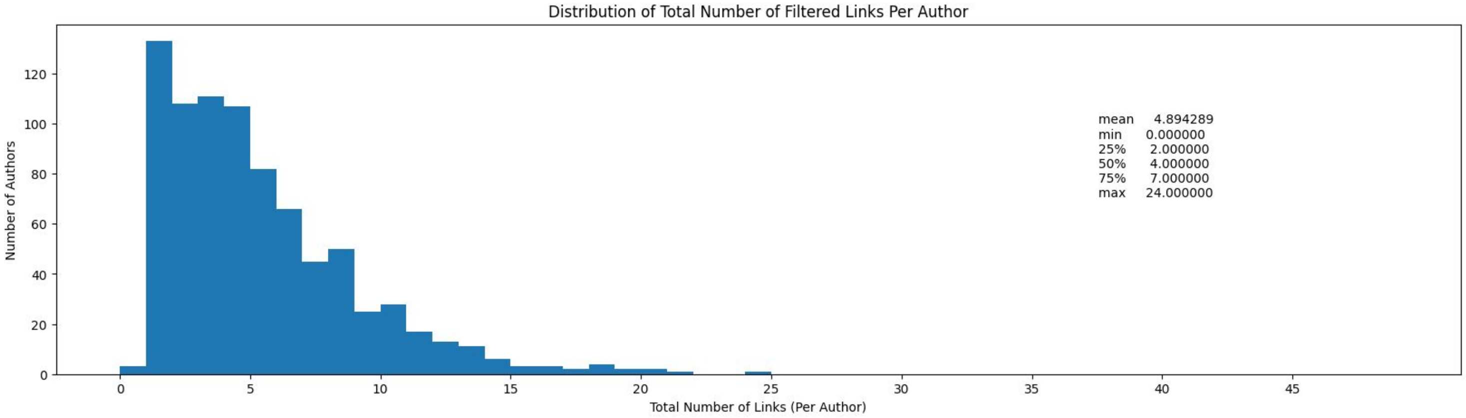
Philadelphia, Pennsylvania, United States · Contact info

500+ connections

Penn Engineering
University *of* Pennsylvania

Google API - Results (Distribution of Total Links Per Author)



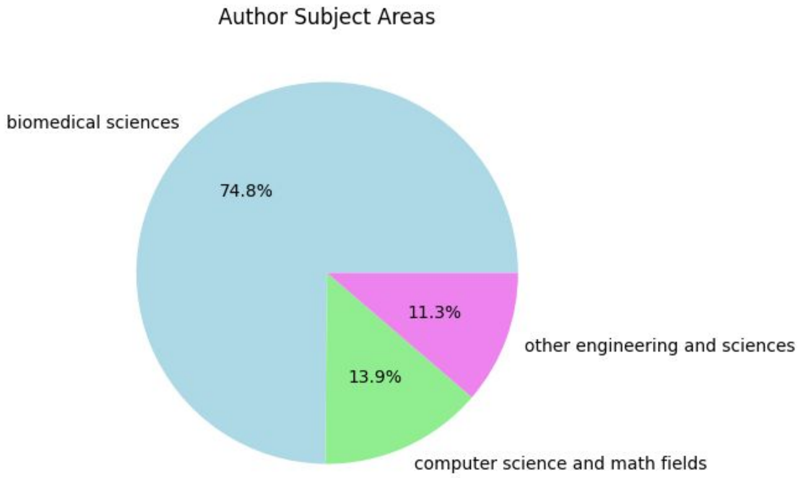


Total number of authors: **887**

After applying our filter, none of our authors have more than 25 total links and our **mean** occurs much earlier, showing how many links we're able to eliminate

Penn Engineering
UNIVERSITY of PENNSYLVANIA

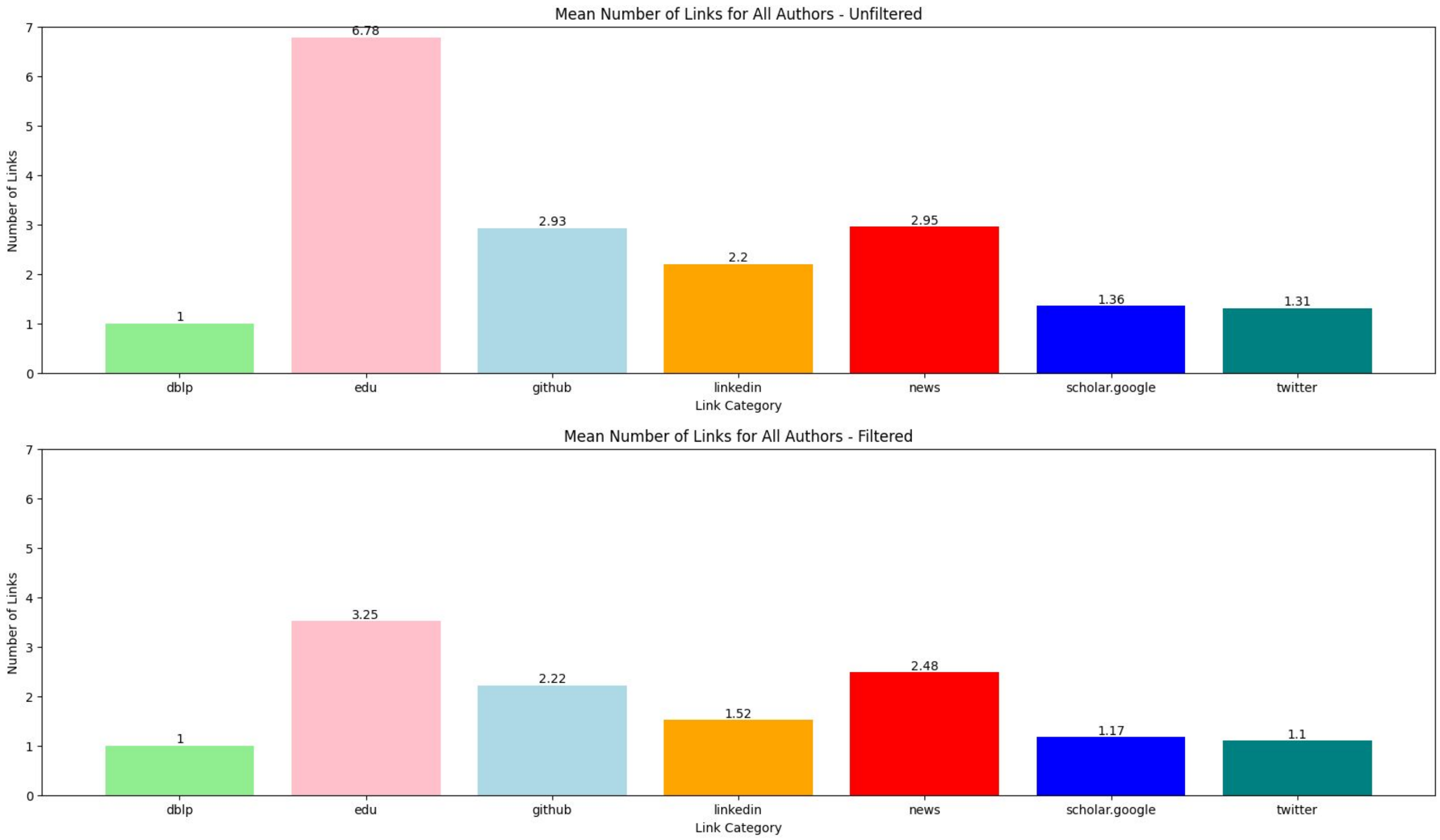
Google API - Results (Subject Area Make-up)



Authors in this sample primarily work in the **biomedical sciences**

Computer sciences and general engineering fields are less represented in the **most cited** authors

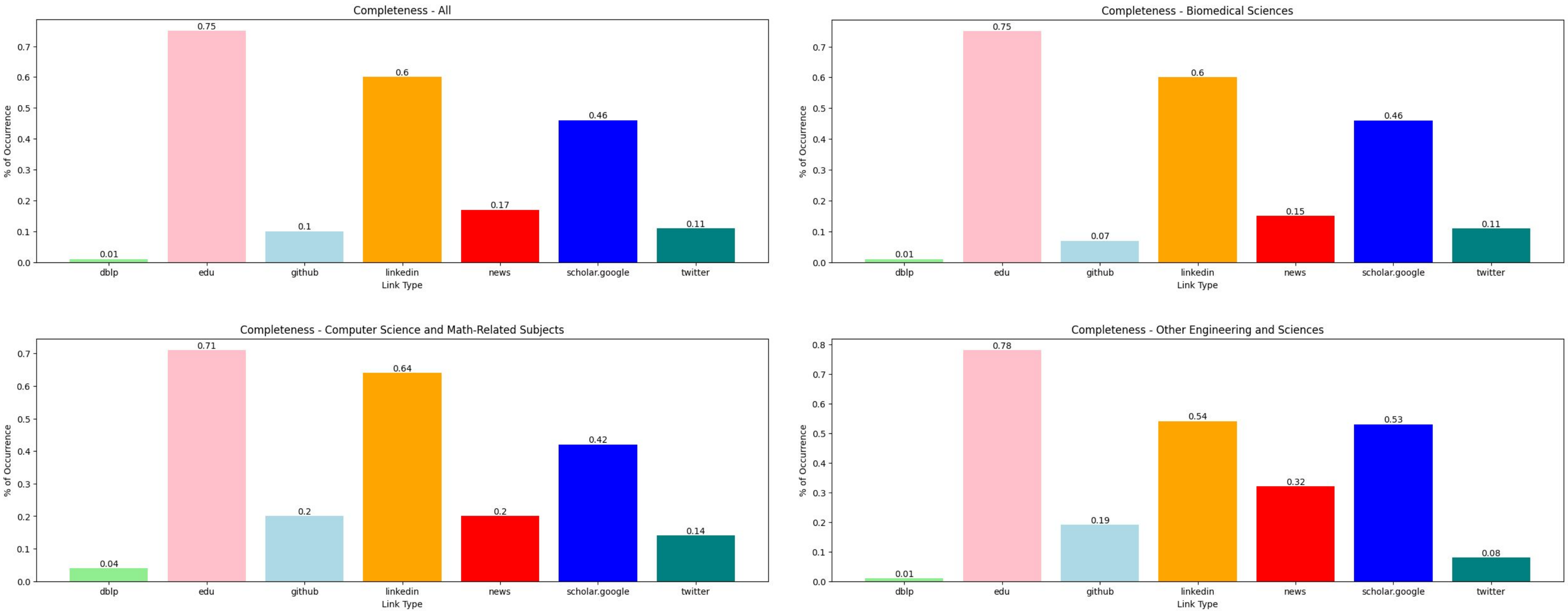
Google API - Results (Mean Number of Links for All Authors)



Average links per category **decreases** in the filtered version (bottom) across all categories

*Notably, **edu** results are cut down with a more than **50%** **decrease**

Google API - Results (Completeness)



- Notice that our results are fairly **consistent** across all of the subject areas
- Computer Science and Math-Related Subjects and Other Engineering and Sciences authors have **more complete Github** results compared to the other categories

Penn Engineering
UNIVERSITY of PENNSYLVANIA

Google API - Capabilities and Limitations

Capabilities:

- Create **search queries** for any author in the IEEE data lake
- Perform a **curated search** for information related to authors of interest
- Fine control over how search and **results filtering/sorting** are performed
- Fast, **asynchronous search** scales to size of author set

Limitations:

- Free-tier Google API query limit restricts search capacity
- Scraping limitations for important domains (**Google Scholar, LinkedIn, Twitter**) impedes sophisticated relevance ranking
- Many authors in the top 1% do not have a GitHub presence

Future Development

The `scholarscraper` package defines two web-scraping models:

Further Developments to this package might include:

- Expanded integration of natural language models for relevance and filtering
  - Cosine Similarity with word embeddings
  - FAISS[6]
  - Fine-tuned BERT model
- Aggregate results across Common Crawl and Google
- Introduce a subroutine to improve handling of Google Scholar and LinkedIn data (ex: SERP API)
- Track API credit limits for the day and pause search/restart at next author in the future

Reflections on the Semester

Improve Efficiency:

- Personnel: parallelized development of project stages:
  (1) link acquisition; (2) model intelligence
- Code: leverage distributed computation with Spark handle big data

Refine Data:

- Author seed data: explore additional data lake variables to curate search queries to individual authors (e.g. subject area)
- Link relevance: implement PageRank-type search to assist removal of bad links

Thank you to the team at IEEE for an excellent semester!

Abhinav Mikaela Rut Ruchika Sam

References

1. Common Crawl (https://commoncrawl.org/)
2. CCNet: Extracting High Quality Monolingual Datasets from Web Crawl Data (arXiv:1911.00359)
3. Facebook FAIR's WMT19 News Translation Task Submission (arXiv:1907.06616)
4. Trafilatura: A Web Scraping Library and Command-Line Tool for Text Discovery and Extraction (doi: 10.18653/v1/2021.acl-demo.15)
5. Semantic Sensitive TF-IDF to Determine Word Relevance in Documents (arXiv.2001.09896v2)
6. Semantic Search with FAISS (https://huggingface.co/learn/nlp-course/chapter5/6?fw=tf)