02807

Computational Tools for Data Science

Identifying Subject Matter Experts on Stack Overflow

Authors: ***Student Numbers:***

[redacted]

Contents

1 Introduction 1
1.1 Scope and Motivation 1
1.2 Sourcing Data 1
1.3 Data Processing 1
1.4 Applied Methods 1

2 Network Construction 1
2.1 Network Analysis 2

3 Topic and Community Detection 3
3.1 LDA & NMF 3
3.2 A-Priori 3
3.3 Network methods 4
3.3.1 Considered methods 4
3.3.2 Louvain algorithm 4
3.4 Community detection results 4

4 Sentiment analysis 4

5 Defining a Subject Matter Expert 5

6 Conclusion 6

A Contributions 7

B Attachments 8

C Community analysis 9

1 Introduction

1.1 Scope and Motivation

Our project analyzes Stack Overflow's user interaction data to identify experts in various programming areas. The scope of the study is thus to analyze Stack Overflow's content through network analysis, representing users and their interactions as nodes and edges. We quantify user engagement and expertise, incorporating sentiment analysis and topic detection for a comprehensive overview of the community.

1.2 Sourcing Data

The data used in this project is a *Stack Exchange Data Dump* accessed in November 2023 via this link: `https://archive.org/details/stackexchange`. The published content is anonymized, and made available for data science projects by the Stack Exchange Network. This extensive dataset, which occupies over 200 GB when converted to CSV files comprises data from Posts, Users, and Comments. Posts include various types like questions, answers, and wiki entries, each with details like creation, activity dates, and scores. Users contain information such as upvotes, downvotes, activity, and profile details. Comments are linked to posts and users, including text, scores, and creation dates. The four primary CSV files used in our project account for approximately 110 GB. For a more comprehensive overview of the data schema documentation, please refer to the documentation [1].

1.3 Data Processing

Preprocessing was essential due to the large size of the original dataset. We filtered for user engagement, keeping users with over 50 contributions (posts, answers, or comments). This criterion, based on the network's activity, narrowed down the user base to about 120,000 active and influential members. However, the dataset remained too large for our computational resources. We further randomly sampled 10% of these users, reducing the size to about 12,000. This step significantly improved our analytical efficiency and maintained a representative sample of Stack Overflow's active community.

To efficiently handle the large volumes of data, we used Dask for parallelized data preprocessing and sentiment calculation. For more information on Dask, please refer to the official documentation[2].

1.4 Applied Methods

To complete this project, we applied the following topics learned during the course:

1. Sentiment Analysis
2. Topic Modelling: A-Priori algorithm, TF-IDF
3. Girvan-Newman algorithm, Spectral clustering

Additionally, concepts outside of the course scope were applied:

1. Louvain algorithm

2 Network Construction

In constructing our network, nodes represented Stack Overflow users in the sampled data. Edges corresponded to user interactions through comments and posts, mapping the exchange of information. This approach enabled a focused analysis of the community's most active participants. Figure 1 provides a

[1] `https://meta.stackexchange.com/questions/2677/`

[2] `https://docs.dask.org/en/stable/`

visualization of the constructed network, illustrating the intricate web of interactions among some of the most active users on Stack Overflow.

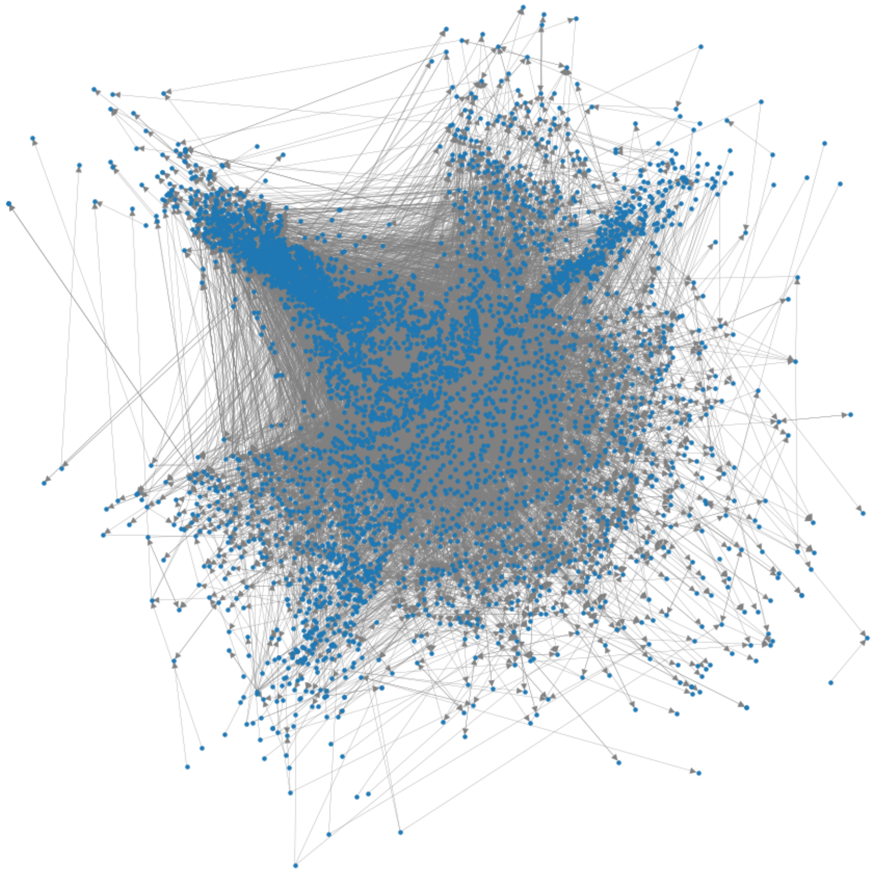

Figure 1: Visual representation of the Stack Overflow network after applying thresholds and sampling. Nodes symbolize active users with over 50 contributions, while edges show their interactions, like answers and comments. Edges from commenters or answerers lead to original posters, reflecting information flow. The graph underscores a dense core of interconnected users and less connected outer nodes, suggesting a scale-free network.

2.1 Network Analysis

The degree distribution of our network suggests a power-law pattern typical of scale-free networks, where a few nodes, or hubs, have many connections, while most have few. On Stack Overflow, these hubs are likely active users with expertise, answering questions on various topics. Their extensive connections suggest they significantly influence and contribute to community knowledge. Figure 2 shows this distribution on a log-log scale, highlighting the network's power-law characteristics.

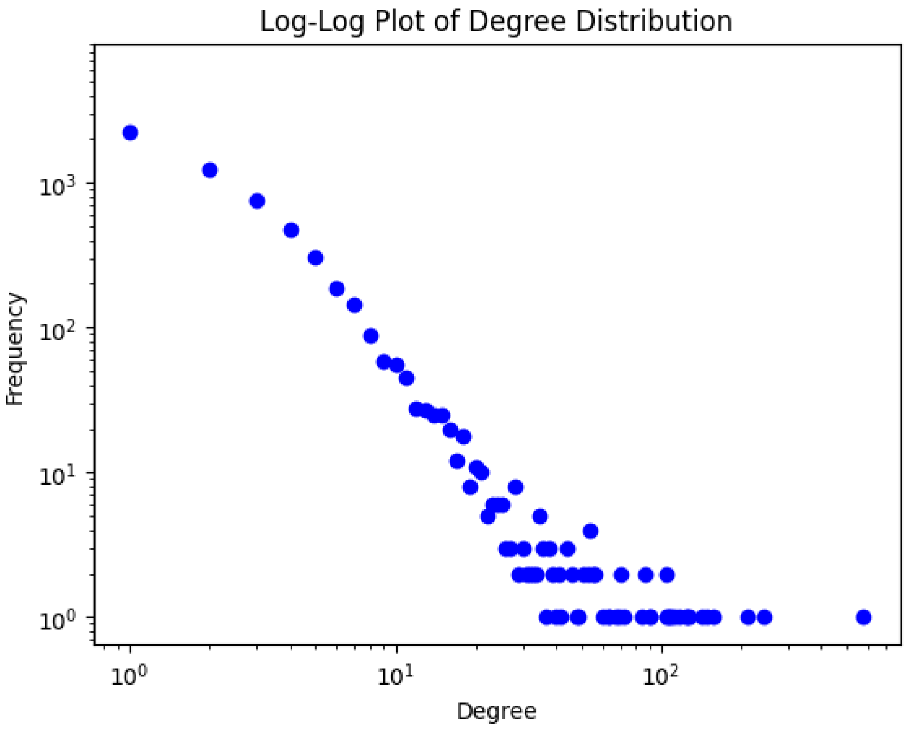

Figure 2: Degree distribution of the network on a log-log scale showing a linear trend that is a signature feature of scale-free networks. The long tail implies a heterogeneity in the network structure, with most users participating in only a few discussions, while a few users have a broad range of influence.

3 Topic and Community Detection

Our objective is to identify the themes, or ’topics’, associated with Stack Overflow posts to detect expert users. Tags assigned by users give a rough idea of the post’s content. Yet, the original dataset contains over 60,000 unique tags, necessitating a strategy to consolidate them into coherent topics. We explored various methods, including latent Dirichlet allocation (LDA) and non-negative matrix factorization (NMF). Despite integrating titles, bodies, and tags with an emphasis on tags, these methods did not yield clear topics, leading us to consider alternative approaches.

3.1 LDA & NMF

Our initial attempts at topic modeling involved using Latent Dirichlet Allocation (LDA) and Non-negative Matrix Factorization (NMF), focusing on the dataset’s of tags and the post contents. However, both LDA and NMF struggled to manage the data’s complexity. Despite integrating titles, bodies, and tags with an emphasis on tags, the methods failed to produce clearly defined topics. We therefore explored methodologies more suited to the dataset’s characteristics.

3.2 A-Priori

We applied the A-Priori algorithm to post tags, organized into itemsets representing each post’s tag collection. The dataset’s tag diversity necessitated lowering the algorithm’s minimum support and confidence (to 20%) to yield results. Although this produced frequent itemsets and association rules, the itemsets were too small for defining substantial topics. These results, however, can validate future findings, as frequent itemsets often contain related topic items.

3.3 Network methods

In search of a way to group tags in bigger sets, we built a network where nodes are individual tags. A link will be made between nodes when both tags are in the same post, increasing the weight of the link corresponding to the number times it occurred. This way, we expect more related tags to have stronger links.

Once the network is built, the idea was to group the nodes into different groups, corresponding to distinct topics in the context of the Python programming language. This process, by definition, includes some arbitrariness, basing on the group's domain knowledge and the necessity to have a smaller number of topics. As learned during the course, a viable approach is community detection, due to the need to label nodes in a network in an unsupervised way.

3.3.1 Considered methods

More methods were tried and evaluated in effort to find the best fitting for our data. The Girvan-Newman algorithm was tested with a random selection of between 200 and 1000 nodes, due to high compute intensity, and resulted in the detection of one dominating community and a handful of communities containing very few tags, making it not feasible for our purposes.

Spectral clustering is adept at uncovering complex community structures not easily detectable through link densities, providing a more global perspective by leveraging the mathematical properties of the network. Starting with 10 partitions - the number of topics aimed to detect, but also trying different values, the results were similarly unfeasible.

3.3.2 Louvain algorithm

The Louvain algorithm quickly identifies communities in large networks by optimizing modularity, a measure of the density of internal community links versus external ones, and can be quite fast and scalable for larger networks. This algorithm has a parameter called *resolution* which determines the sizes of the detected communities. If the resolution parameter is set below 1, the algorithm tends to favor larger communities; setting it above 1 favors smaller communities. This method was suitable for our project, with chosen resolution of 1.0. that conclusion was reached after evaluating resolution values from 0.5 to 2.5 in 0.1 increments.

3.4 Community detection results

The 10 detected significant communities - the ones formed by more than 400 tags - and their attributes are presented in appendix C. Within each community there's most used tags, total tags and their instances, as well as general descriptions. Those are the communities - topics, that we aim to be able to find experts in.

4 Sentiment analysis

As part of measuring to what extent other users are satisfied with our potential subject matter experts, we performed sentiment analysis on comments. Each comment to an answer authored by one of the users in our data was analyzed - stopwords were removed, and the text was lemmatized. Then, each comment was appointed a compound polarization score, using vader sentiment analysis. Mean sentiment score of comments to all answers, as well as mean sentiment score per topic was added to the users dataframe.

5 Defining a Subject Matter Expert

Our objective is to establish a comprehensive metric for assessing subject matter expertise. We formulated equations that encapsulate key aspects of user engagement and their contributions' impact across different subjects. These metrics are individually calculated for answers (A), comments (C), and posts (P), as shown below:

$$S_A^{u,i} = N_A^{u,i} \cdot (Q^{u,i} + \alpha \cdot \frac{l_A^{u,i}}{v_A^{u,i}}), \qquad S_C^{u,i} = N_C^{u,i} \cdot \frac{l_C^{u,i}}{v_C^{u,i}}, \qquad S_P^{u,i} = \frac{l_P^{u,i}}{v_P^{u,i}}$$

where:

$S_X^{u,i}$ = score for X type of contribution by user u on topic i
$N_X^{u,i}$ = number of X type of contribution by user u on topic i
$Q^{u,i}$ = average sentiment of comments to user u's contributions on topic i
$l_X^{u,i}$ = sum of score of X type of contribution by user u on topic i
$v_X^{u,i}$ = sum of views of X type of contribution by user u on topic i
α = coefficient used to weight the ratio of likes and views differently than the $Q^{u,i}$

To synthesize these individual metrics into a unified score for comparative analysis across topics, we introduce weighting coefficients. These coefficients are adjustable to prioritize certain metrics over others, alongside the α coefficient in the initial equations.

$$Score^{u,i} = S_A^{u,i} \cdot w_A + S_C^{u,i} \cdot w_C + S_P^{u,i} \cdot w_P$$

where

w_X = coefficient for X type of contribution score

Using this scoring formula, we can rank users in each topic. Our plan is to identify the top X percentile of users per topic, balancing the need for a significant expert presence with the exclusivity of the 'expert' designation. This ensures a rich contribution pool, maintaining the value and distinction of the expert category.

To ensure substantial expert contribution, after retrieving the top percentile of users, we evaluated how many of all processed posts had activity from those users, as visible in Figure 3. The initial rapid incline of the plot line is concurrent with our expectation that the experts are hubs in the network, participating in many of the posts. However, since the participation measurement takes into consideration number of posts that have at least one contribution by an expert, the curve eventually flattens as more users contribute to the posts that were already accounted for (as well as being users that we don't expect contribute as much). Based on this analysis, we recommend the value of X to be between 5 and 20.

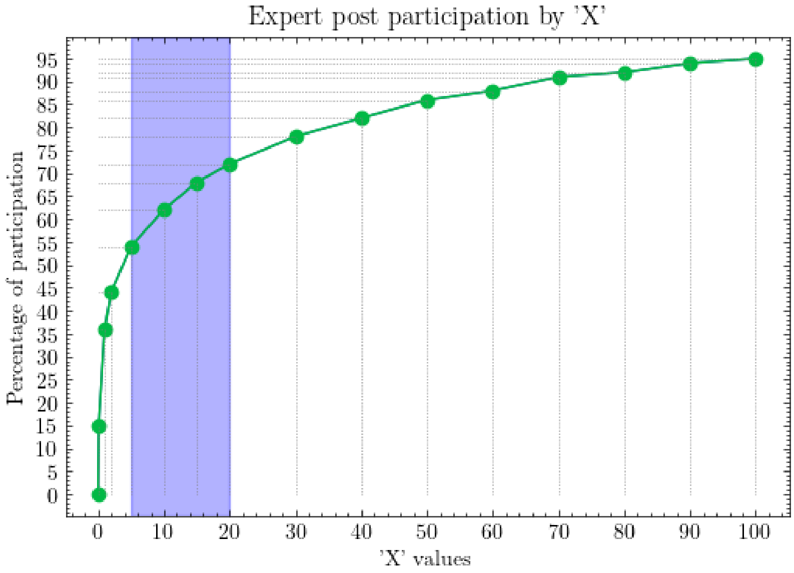

Figure 3: Line plot representing the effect of X on the amount of posts and their corresponding answers and comments that have an expert contribution.

6 Conclusion

In our project we have demonstrated a framework for identifying subject matter experts within the Stack Overflow community. By leveraging advanced network analysis, sentiment analysis, and community detection algorithms, we have successfully highlighted users who not only actively contribute but also offer valuable insights and solutions in their respective fields. The performed analysis shows the complexity and scale of data, yet providing a way to recognize users with majority of well-rated contributions to different topics. While our current methodology is robust, it could be further refined by enhancing data accuracy, especially in addressing missing or mismatched information. Additionally, experimenting with a wider range of weighting factors in our algorithm could yield even more precise results. This framework, thus, stands not just as a tool for expert identification but also as a testament to the potential of data-driven insights in understanding and harnessing the power of online communities.

A Contributions

All members equally contributed to the project, and many sections were authored together. More detailed contributions that point to authors of majority of each topic can be described as follows:

1. Preprocessing data [redacted]
2. Network construction [redacted]
3. Network analysis [redacted]
4. Sentiment analysis [redacted]
5. Topic detection [redacted]
6. Defining subject matter experts [redacted]

B Attachments

Alongside this report we hand in a run-through of our methodology. Please refer to our Jupyter Notebook: "*subject_matter_experts.ipynb*".

Due to the size of the data and multiple operations performed on it, the main explainer notebook will not run. All the sectioned notebooks that work on pickle files can be found in our Github repository. Particular pickle files can be provided on request, as they are too big to include in the repository.

The main provides the reader with a thorough overview of our project solutions. Specifically, we will walk through the following;

Data

- Downloading and parsing data from the source
- Reading parsed data and typecasting the data
- Filtering relevant posts, comments and users

Topic Modelling

- TF-IDF, Latent Dirichlet Allocation (LDA) and Non-negative Matrix Factorization (NMF)
- A-Priori and frequent itemsets

Network

- Creating the network

Sentiment Analysis

- Sentiment of comments towards user answers

Subject Matter Experts

- Defining the Expert Metric
- Top n result

Network Analysis

- Network Algorithms

C Community analysis

Community 1: Python Data Processing and Analysis

- **Top 10 Tags**: python-3.x, pandas, numpy, dataframe, python-2.7, list, dictionary, regex, json, arrays
- **Total Tags in Community**: 4712
- **Total Tag Instances**: 2,156,967
- **Theme Description**: Focused on data processing and analysis in Python, with prominent data handling libraries and structures.
- **Non-Congruent High-Count Tags**: None

Community 2: Python Web Frameworks and Technologies

- **Top 10 Tags**: django, flask, javascript, django-models, mysql, sqlalchemy, django-rest-framework, docker, django-views, postgresql
- **Total Tags in Community**: 5959
- **Total Tag Instances**: 981,845
- **Theme Description**: Centered around Python web development, covering Django and Flask, and integrating various backend and frontend technologies.
- **Non-Congruent High-Count Tags**: None

Community 3: Python in Diverse Operating Systems and Environments

- **Top 10 Tags**: linux, pip, jupyter-notebook, windows, c++, pygame, multithreading, macos, java, pycharm
- **Total Tags in Community**: 9885
- **Total Tag Instances**: 973,462
- **Theme Description**: Covers Python usage in various operating systems and environments, with a focus on OS-specific questions, package management, and IDEs.
- **Non-Congruent High-Count Tags**: c++, java (indicating interdisciplinary programming)

Community 4: Python in Machine Learning and Image Processing

- **Top 10 Tags**: tensorflow, opencv, keras, machine-learning, scikit-learn, deep-learning, pytorch, image-processing, image, nlp
- **Total Tags in Community**: 2672
- **Total Tag Instances**: 536,960
- **Theme Description**: Revolves around machine learning and image processing, with an emphasis on frameworks like TensorFlow, Keras, and PyTorch, and topics like NLP and image processing.
- **Non-Congruent High-Count Tags**: None identified

Community 5: Python for Data Visualization and GUI Development

- **Top 10 Tags**: matplotlib, tkinter, pyqt, pyqt5, plot, r, user-interface, seaborn, plotly, wxpython
- **Total Tags in Community**: 2604
- **Total Tag Instances**: 429,182
- **Theme Description**: Focuses on data visualization (Matplotlib, Seaborn, Plotly) and GUI development (Tkinter, PyQt), with some overlap into the R programming language.
- **Non-Congruent High-Count Tags**: 'r' indicates some cross-language integration

Community 6: Web Scraping and Automation with Python

- **Top 10 Tags**: selenium, html, web-scraping, beautifulsoup, python-requests, selenium-webdriver, scrapy, xml, xpath, selenium-chromedriver
- **Total Tags in Community**: 2098
- **Total Tag Instances**: 395,626
- **Theme Description**: Dedicated to web scraping and browser automation, featuring tools like Selenium, BeautifulSoup, and Scrapy, along with web technologies like HTML and XML.
- **Non-Congruent High-Count Tags**: None identified

Community 7: Python Scripting and System Interaction

- **Top 10 Tags**: pyspark, amazon-web-services, apache-spark, azure, aws-lambda, amazon-s3, boto3, apache-spark-sql, amazon-ec2, elasticsearch
- **Total Tags in Community**: 1919
- **Total Tag Instances**: 190,485
- **Theme Description**: Focuses on Python scripting and interaction with system-level components, particularly in cloud services and big data processing. It includes a strong emphasis on cloud computing platforms like AWS and Azure, and big data technologies such as Apache Spark and Elasticsearch.
- **Non-Congruent High-Count Tags**: None identified

Community 8: Python Object-Oriented Programming and Testing

- **Top 10 Tags**: class, oop, pytest, unit-testing, inheritance, object, testing, python-sphinx, python-unittest, mocking
- **Total Tags in Community**: 744
- **Total Tag Instances**: 136,251
- **Theme Description**: Focused on object-oriented programming (OOP) in Python, covering aspects like classes, inheritance, and objects, as well as testing methodologies including unit testing and mocking.
- **Non-Congruent High-Count Tags**: None identified

Community 9: Python and Google Cloud Services

- **Top 10 Tags**: google-app-engine, google-cloud-platform, airflow, email, google-bigquery, google-cloud-datastore, google-cloud-storage, google-api, go, google-drive-api
- **Total Tags in Community**: 953
- **Total Tag Instances**: 109,859
- **Theme Description**: Revolves around Python's integration with Google Cloud services and platforms, including App Engine, BigQuery, and Cloud Storage, with some overlap into Google's Go language.
- **Non-Congruent High-Count Tags**: 'go' suggests some focus on Google's Go language

Community 10: Python in Text Processing and Document Management

- **Top 10 Tags**: unicode, pdf, encoding, utf-8, encryption, cryptography, character-encoding, ascii, python-docx, pypdf
- **Total Tags in Community**: 813
- **Total Tag Instances**: 70,346
- **Theme Description**: Concentrates on text and character encoding (Unicode, UTF-8, ASCII), document management (PDF processing with Python-docx, PyPDF), and aspects of encryption and cryptography.
- **Non-Congruent High-Count Tags**: None identified