Network Science Investigation on Social Dynamics

Alessia Petracin[1], Saima Sharleen al Islam[2]

[1]alessia.petracin@studenti.unitn.it

[2]saimasharleen.islam@studenti.unitn.it

***Abstract*— This study introduces a systematic approach for examining and representing email conversation records by leveraging various technologies, such as Kafka, Apache Spark, PostgreSQL, SQLite, the Gmail API and the Girvan-Newman algorithm. The objective is to extract meaningful observations from patterns of communication and present them through interactive visualization using D3.js. The Girvan-Newman method was employed to detect communities within a communication network, thereby uncovering latent patterns and clusters. The approach, conclusions, and outcomes of the project have been thoroughly documented. The application of the Girvan-Newman algorithm exemplifies the project's novel methodology for revealing complex communication dynamics.**

***Keywords*— Social Networks, Cluster Analysis, Social Dynamics, Communication Analysis, Email Interactions, Kafka, Apache Spark, PostgreSQL, Network Analysis, Girvan-Newman Algorithm, Community Detection, Interactive Visualization, Communication Patterns, Cluster Identification, Data Analytics**

I. Introduction

In contemporary times characterized by digital communication, the use of email interactions has become an essential and integral element within the realm of current communication landscapes. The large volumes of data produced via email communications possess a significant amount of valuable information regarding patterns of human interaction, structures within communities, and concealed connections[3]. The use of these observations asks for state-of-the-art technologies as well as the execution of advanced techniques for data retrieval, storage, processing, analysis, and visualization. This research offers a thorough examination of several approaches, revealing significant insights derived from email exchange records and displaying them through interactive visualizations[1].

The ongoing evolution of digital communication has underscored the continued significance of email contacts as a vital means of correspondence across several sectors. Nevertheless, the considerable amount of data produced as a result of these exchanges poses a substantial obstacle in effectively using the inherent patterns and connections[8]. The comprehension of the concealed communication dynamics present in these interactions can provide organisations, researchers, and individuals with significant insights[2]for the purposes of decision-making, community analysis, and relationship management. In order to tackle this difficulty, the project adopts a multidisciplinary methodology that incorporates many technologies such as the Gmail API[11], Kafka[10], Apache Spark, PostgreSQL[9] and SQLite[12]. These technologies are used to gather, store, manipulate, examine, and present data pertaining to email correspondence.

II. System Model

A. *System architecture*

In this undertaking, we used a variety of technological stack, incorporating technologies such as the Gmail API, Kafka, PostgreSQL, SQLite, Spark, and visualisation libraries to develop a comprehensive system capable of managing the complex nuances of communication data. The fundamental basis of our research is in the use of the SNAP Email-Eu-core dataset, which acts as the bedrock for our datacentric investigation. Expanding upon our research, we have developed a complete system model that represents the complex data flow and operations. The approach consists of three separate pipelines, each serving a specific purpose and designed to manage the diverse phases of data transformation and analysis. The framework comprises a Data Production Pipeline, Stream Processing Pipeline, and Data Consumption and Analysis Pipeline, all of which play a role in advancing the project as a whole.

1 Data Production - The system model is initiated by the Data Production Pipeline, which starts with the selection of the SNAP Email-Eu-core dataset. The dataset undergoes a process of data extraction and transformation, resulting in its precise conversion into CSV format. Following this, the data undergoes a seamless ingestion process into Kafka, establishing the foundation for subsequent processing.

An alternative pathway provided by the current project is through the Gmail API. Through the addition of this feature, people can retrieve the data directly from their own email box. However, this second approach presents a number of limitations in the quality of the data, as it only refers to one person's ego network. By contrast, the possibility to study one's own ego network's characteristics offers an additional opportunity to test theories and visualizations on a more tangible reality for a potential researcher.

2 Stream Processing – Subsequently to the data production, the baton is transferred to the Stream Processing Pipeline, wherein the Kafka stream is activated. The incoming data is subjected to a sequence

of crucial preparation procedures to ensure its cleanliness and accuracy. The pre-processed data is subsequently

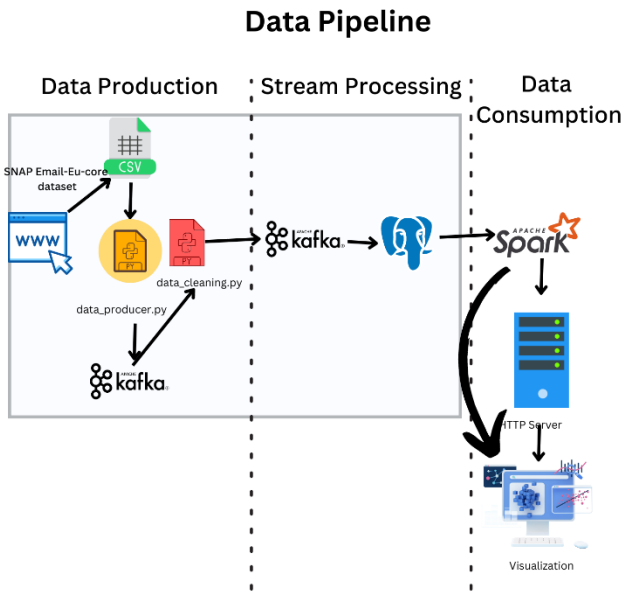

Fig. 1 Data pipeline

directed to a PostgreSQL database, where it is stored through a procedure of cleansing and organising.

An additional storage option is also possible, through the SQLite engine, more suitable for small-scale research, where resource constraints are a concern.

3 Data Consumption - The Data Consumption and Analysis Pipeline incorporates the PostgreSQL database as its central component. The pre-processed data is obtained, enabling in-depth analysis facilitated by Spark. The present study explores sophisticated network analysis algorithms, such as the Girvan-Newman algorithm, with the aim of revealing latent communication patterns and exposing buried insights. By leveraging a powerful amalgamation of Python libraries such as Matplotlib and NetworkX, in conjunction with the capabilities offered by JavaScript libraries like D3.js, we are able to create compelling visual representations that enhance the interpretability and depth of our analytical findings. The culmination of the visualization process is the development of a web-based dashboard, which is hosted by an HTTP server.

B. Technologies

Data is provided either through a pre-existing database or, alternatively, through the Gmail API, directly storing the data into a relational database. The various technologies employed collaborate harmoniously to establish a cohesive and effective process, enabling the extraction of intricate insights from unprocessed communication data. Through this approach, one is capable of conducting a thorough analysis of email conversations, thereby extracting significant insights regarding patterns and dynamics in the network.

1 Gmail API - The Gmail API[11] (Application Programming Interface) is a set of tools and protocols developed by Google that allows developers to interact programmatically with Gmail, Google's email service. It enables developers to integrate Gmail features, such as sending and receiving emails, managing labels, and accessing various aspects of a user's mailbox, into their own applications, services, or websites. The option to use the Gmail API enables users to retrieve data directly from their own email box, therefore making it possible to provide insights into one's own personal communication network.

2 Kafka - Kafka[7] is a distributed event streaming platform that has been developed by the Apache Software Foundation and is available as an open-source solution. This technology is employed in the context of high-performance data pipelines, streaming analytics, data integration, and applications that are of utmost importance. The aforementioned characteristics render Kafka a highly suitable instrument for the management of real-time data streams.

3 PostgreSQL - PostgreSQL[6] is an open-source object-relational database system with over 35 years of active development. It is known for its reliability, feature robustness, and performance. PostgreSQL offers advanced features such as complex SQL queries, transactional integrity, and concurrency control. It is employed in the context of our project due to its scalability, making it possible to use it both for small researches and large enterprise systems.

4 SQLite – SQLite[12] is a self-contained, serverless, and lightweight relational database engine that provides a seamless way to manage and store structured data within applications. Designed for simplicity and efficiency, SQLite is embedded directly into software, eliminating the need for a separate database server. It is a popular choice for mobile apps, desktop applications, and small-scale projects. Despite its minimal footprint, SQLite supports essential database features, including transactions, indexing, and querying, while maintaining data integrity and reliability. The addition of SQLite into the current project serves as a tool where resource constraints are present.

5 Apache Spark - The Apache Spark[10] framework serves as the foundation for our Data Consumption and Analysis Pipeline, providing a distributed and adaptable data processing engine. By utilising the sophisticated analytical features of Spark, we employ network analysis tools such as the Girvan-Newman algorithm to uncover concealed patterns of communication dynamics. The use of Spark's in-memory processing and parallelized computation facilitates the acceleration of analysis, hence enabling the rapid derivation of insights.

6 Interactive Visualization (D3.js, Matplotlib, NetworkX) - Our methodology expands beyond basic analysis, encompassing the domain of interactive

visualisation. By leveraging a powerful combination of JavaScript libraries such as D3.js and Python tools like Matplotlib and NetworkX, we are able to create visually captivating representations that bring our research findings to life. D3.js enables the creation of dynamic and interactive visualisations for web-based platforms, while Matplotlib and NetworkX provide comprehensive analysis of network structures and dynamics.

7 HTTP Server - In order to disseminate our insights to a wider audience, we utilise a web-based dashboard that is hosted by an HTTP server. Flask, a lightweight web framework, functions as the foundational element of our interactive presentation, facilitating the smooth integration of our data analysis and visualisations into a user-friendly interface.

Fig. 2 Web Server UI

III. Implementation

The figure illustrates the installation of the five components for data creation and streaming.

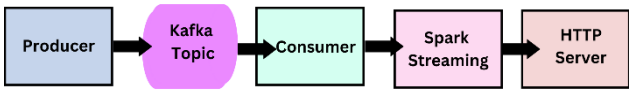

Fig. 3 Flow of the architecture

Gmail RESTful API for Data Retrieval. As previously mentioned, the data can be either retrieved from a pre-existing database or provided through the Gmail API. To achieve this result, a connection has to be established to the Gmail API using OAuth 2.0 Authentication, returning a service object that can be used to make API requests. At this point, the emails that match a specified query are retrieved, within a given date range. Each email sender and receiver is associated to a unique ID for privacy reasons and the resulting IDs are stored in a PostgreSQL database, within a specified table.

Integration of Kafka for Data Ingestion. The data pipeline was launched by leveraging the functionalities of Kafka. The conversation logs retrieved from the Snap email dataset were smoothly imported using a customised Kafka producer. The publish-subscribe architecture devised by Kafka facilitated the seamless and dependable transmission of data streams, enabling the real-time recording and transmission of email interactions.

PostgreSQL Database Management. PostgreSQL plays a significant role in data management throughout the Stream Processing Pipeline. An organised schema was devised to accommodate the email data, and the Python adaptor psycopg2 was utilised to facilitate interaction with the PostgreSQL database. The implementation permitted seamless movement of data from Kafka to the database, hence ensuring the preservation of data integrity and optimising storage efficiency.

SQLite for alternative storage. An SQLite database is created, defining a table structure and specifying columns names. The data either retrieved from a CSV file or an API.

IV. Results

The analysis conducted in our research used the communication network extracted from the email EU dataset, which yielded a comprehensive set of 265,214 nodes and 365,570 edges. The numerical value "9" derived from the degree distribution corresponds to the count of individuals (nodes) inside the network who possess a particular degree (quantity of connections).

Clustering coefficient: The clustering coefficient is a statistical measure that offers useful insights into the underlying structural patterns inside a network. The metric quantifies the extent to which nodes demonstrate clustering tendencies or establish interconnected clusters. It can be observed that Node 0 exhibits a clustering coefficient of roughly 0.0667, suggesting that it establishes links with its neighbouring nodes, albeit the neighbours themselves do not display a significant level of interconnectivity. The clustering coefficient for ode 4 is calculated to be 0.0116. Node 4 exhibits a clustering coefficient of approximately 0.0116, suggesting that a subset of its neighbouring nodes establish connections amongst themselves, hence contributing to the phenomenon of local clustering. The clustering coefficient of Node 11 is calculated to be 0.1291. Node 11 exhibits a notable clustering coefficient of roughly 0.1291, indicating a strong interconnectivity among its neighbouring nodes, resulting in the formation of a cohesive cluster. The clustering coefficient of Node 48 is 0.0008. The average clustering coefficient of the network is approximately 0.067, indicating a moderate tendency for nodes to form interconnected groups or clusters.

Centrality Measures: Centrality measures help us understand who the most important people in the network are. We are using a directed graph as we are interested emails being sent and received.

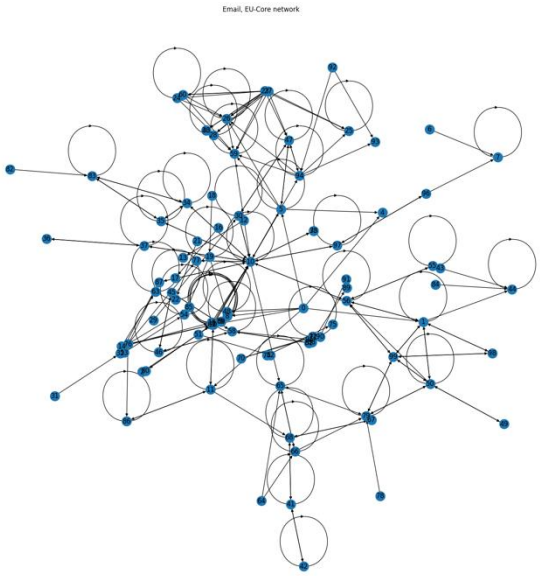

Fig. 4 Centrality Measures

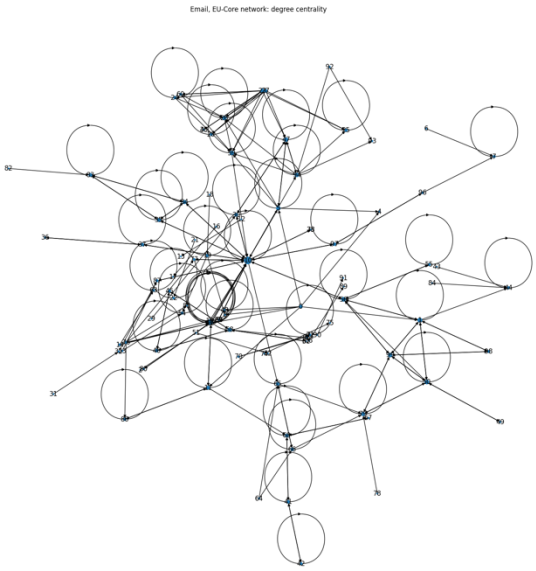

Fig. 5 Degree Centrality

We can see that there are two densely connected networks and many connections between them. Through an analysis of the extensive quantity of emails transmitted and received, we have discerned an individual who distinguishes themselves by exhibiting the highest degree of engagement. The individual in question, possessing an index value of 9, has exhibited a notable degree of involvement, signifying a substantial level of participation and interaction within the network.

Degree centrality: The concept of evaluating the significance of an individual based on the amount of connections they possess, also known as "Degrees," is referred to as degree centrality. The following list presents the top 3 individuals with the highest number of connections, arranged in ascending order.

TABLE-I
DEGREE CENTRALITY

Node	Degree Centrality
10	0.3333
1	0.1212
50	0.1111

This network can be visualised with most size proportional to the importance according to degree centrality. This network can be visualised with most size proportional to the importance according to degree centrality.

Community Detection: The Girvan-Newman algorithm, which is a commonly employed technique for network community detection, employs a top-down approach to progressively divide a network into distinct communities through repeated steps.

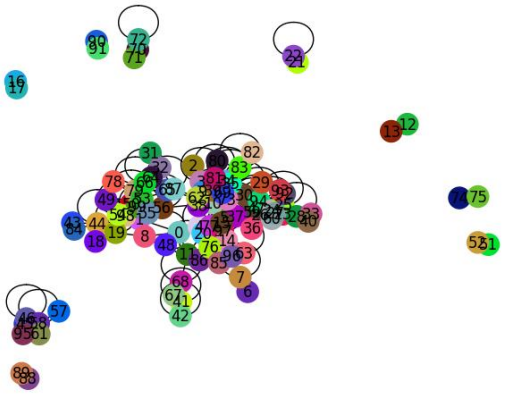

Fig. 6 after removing maximum betweenness values, the community splits into several parts

When we calculate the same split, we get the following visualization:

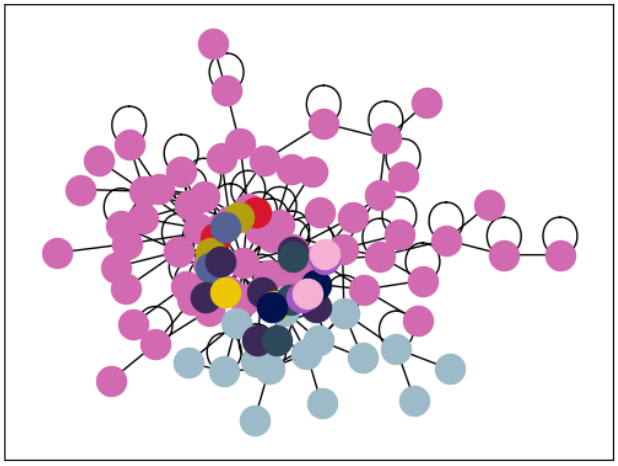

Fig. 7 Calculating from the same split

Dunbar's number: The network's average degree is 3.38, indicating that individuals on average interact with about 3.38 other individuals. The average clustering coefficient is 0.15, suggesting moderate local interconnectedness. Dunbar's Number is 150, a theoretical limit on the number of stable relationships a human can maintain. As the network size aligns with Dunbar's Number, it supports the notion that the network's social structure is in line with the proposed cognitive limit.

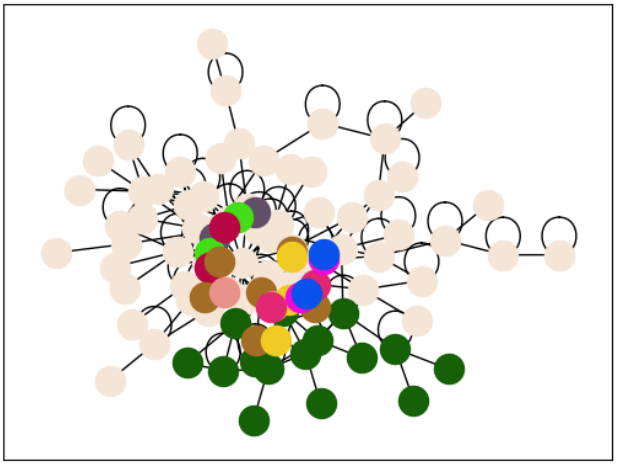

Fig. 8 network size is consistent with Dunbar's number

Discussion

A conflict arose between PostgreSQL and Apache Spark, however, a resolution was achieved by implementing a meticulous tuning of their individual connection settings and optimisation parameters. The process entailed modifying the configuration parameters of the database connection pool, fine-tuning the memory allocation of the Spark cluster, and optimising the execution plans of queries. This ultimately facilitated a seamless and effective analysis of the dataset.

While this study presents a systematic approach for analyzing email communication records through a diverse set of technologies, there are certain limitations that should be acknowledged. Firstly, the application of the Girvan-Newman algorithm for community detection assumes that communities within the network are well-defined and distinct, potentially overlooking more nuanced relationships. Additionally, the study's findings are based on the specific dataset used, and results may vary when applied to different datasets with varying communication patterns.

V. **Conclusions**

In conclusion, this study has presented a novel approach for extracting insights from email conversation records by using a fusion of various technologies. The network analysis, conducted using the Girvan-Newman algorithm, has unveiled underlying communication patterns and clusters. The correlation between network size and Dunbar's Number provides empirical evidence for the cognitive constraint on human interactions. The significance of optimisation in achieving smooth data analysis is exemplified by the thorough configuration tweaks made to resolve conflicts between PostgreSQL and Apache Spark. Through the utilisation of various methodologies and technologies, the present study presents a valuable conceptual framework for conducting a thorough analysis and visual representation of intricate communication patterns within networks.

REFERENCES

[1] Kathleen M Carley. 2018. ORA: A Toolkit for Dynamic Network Analysis and Visualization. In Encyclopedia of Social Network Analysis and Mining. Springer, 1–12.

[2] Jana Diesner and Kathleen M Carley. 2015. Communication networks from the Enron email corpus: “It’s always about the people. Enron is no different”. Computational & Mathematical Organization Theory 21, 3 (2015), 300–328.

[3] Sanby Lee. 2016. MailTronome: Visualizing the rhythm of social interactions through email communication patterns. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. 1–6.

[4] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[5] Andrew McCallum, Xuerui Wang, and Andrés Corrada-Emmanuel. 2007. Topic models for email processing. In Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. 104–112.

[6] Bruce Momjian. 2001. PostgreSQL: Introduction and Concepts. "Addison-Wesley Professional".

[7] Neha Narkhede, Gwen Shapira, and Todd Palino. 2017. Kafka: The Definitive Guide: Real-Time Data and Stream Processing at Scale. Ö’Reilly Media, Inc. ¨ .

[8] John R Tyler, John C Tang, and Karen McCall. 2003. Can automated text analysis be useful in survey research? Proceedings of the International Conference on Survey Research Methods (2003).

[9] Xingbo Wang, Jianben He, Zhihua Jin, Muqiao Yang, Yong Wang, and Huamin Qu. 2021. M2Lens: Visualizing and Explaining Multimodal Models for Sentiment Analysis. IEEE Transactions on Visualization and Computer Graphics 28, 1 (2021), 802–812.

[10] Matei Zaharia, Reynold S. Xin, Patrick Wendell, Tathagata Das, Michael Armbrust, Ankur Dave, Xiangrui Meng, Josh Rosen, Shivaram Venkataraman, Michael J. Franklin, et al. 2016. Apache Spark: A Unified Engine for Big Data Processing. Commun. ACM 59, 11 (2016), 56–65.

[11] Panoramica dell’API Gmail | google for developers (2023) Google. Available at: https://developers.google.com/gmail/api/guides?hl=it (Accessed: 02 August 2023).

[12] Gaffney, K.P. *et al.*, 2022. ‘SQLite’, Proceedings of the VLDB Endowment, 15(12), 3535–3547. doi:10.14778/3554821.3554842.

[13] The code and documentation for the analysis conducted in the present study can be found on the authors’ shared GitHub repository at the link https://github.com/saimasharleen/BDT2023-Group11.