**Influence of Popularity on Sales**

A picture containing colorfulness, text, map

Description automatically generated

Figure 1 - Graph Visualization, author image

FS2023: 63091 Social Media Analytics

**Sophie Caroni & Emmanuel Cazzato & Quentin Nater** [sophie.caroni@unifr.ch](mailto:sophie.caroni@unifr.ch) & [emmanuel.cazzato@unifr.ch](mailto:emmanuel.cazzato@unifr.ch) & [quentin.nater@unifr.ch](mailto:quentin.nater@unifr.ch)

**Table of Contents**

[1 Introduction 1](#_Toc136012532)

[2 Loading data and pre-processing 1](#_Toc136012533)

[3 Graph exploration and visualization 2](#_Toc136012534)

[4 Data persistence 3](#_Toc136012535)

[5 Data enrichment 3](#_Toc136012536)

[6 Community detection 3](#_Toc136012537)

[6.1 Evaluation of the community detection quality 4](#_Toc136012538)

[7 Results 4](#_Toc136012539)

[7.1 Technical results 4](#_Toc136012540)

[7.2 Relating to the hypotheses 5](#_Toc136012541)

[8 Conclusion 5](#_Toc136012542)

**Table of Figures**

[Figure 1 - Graph Visualization, author image 1](#_Toc136005254)

1. Introduction

In recent years, social media has become a vast source of valuable data for businesses and researchers alike. The ability to extract insights and make informed decisions based on this data has led to the emergence of social media analytics as a powerful tool. In our project, we focus on Amazon product data (<http://snap.stanford.edu/data/amazon-meta.html>) and exploit the graph representing its network to gain insights into the role of popular nodes, different communities, and edge connections.

The Amazon dataset provides us with a rich collection of information about products, including their Amazon Standard Identification Number (ASIN), category, salesrank, review, and similar products. We aim to discover patterns and relationships to determine if the central nodes of communities are in the top rank sales.

The first step in our analysis involves constructing a sampled graph representation of the dataset to do some graph exploration and visualization. Each product ASIN is captured as node and the list of products similar to it generates its trailing edges. This directed graph allows us to detect communities and analyze the relation between products.

We aim to answer technical questions such as: What is the better way to sample our graph? Is our design graph fully connected? Is noise corrupting the graph? What is the better algorithm to produce the most efficient community detection? How to evaluate this algorithm?

These technical answers will finally lead to business questions such as: Which product has the highest degree of influence within their community? Are the popular nodes of each community the most sold?

Through our project, we hope to provide a comprehensive analysis of the popular nodes and their community detection. This research has the potential to inform publishers, authors, and marketers about effective strategies for promoting products, and understanding the impact of social media recommendation on sales.

The project has been implemented on GitHub. You can find the repository at this link: <https://github.com/qnater/InfluenceOnSales>

1. Loading data and pre-processing

We used the Amazon metadata dataset from 2006. It contains comprehensive information for 548’552 products, and for each of them a list of similar products (even outside the dataset itself). This leads to a total of 719’150 nodes. Each product node is connected by an outgoing edge to its similar ones, resulting in our directed graph having 1’788’712 edges.

Considering the large size of the dataset, we worked on a sampling algorithm to preprocess and clean it, in order to manage the computational requirements efficiently.

First, we loaded the original text file and exported the relevant data only into a smaller text file. During this process, we constructed the original graph by using all ASIN as nodes and their similarity list as edges.

Then, we cleaned the noise and reduced the size of the dataset by calling iteratively four algorithms. The loop does the following, until the desired size is reached: while it is impacting the graph, it removes the isolated, not in-edged and out-edged nodes, otherwise the nodes with the lowest degree possible.

In this way, we retained the greatest number of relevant nodes for community detection and the lower number of less impactful ones. The resulting sampled graphs were of specific desired sizes with the a managed change in cluster coefficient and mean degree (compared to the initial one).

Finally, we have created features that will export and quickly reconstruct the sampled graph. The motivation to do so lies in running only one time the long sampling algorithm and then recall with a small amount of time the exported graph for any future utilization.

In the table below, we can find the conclusion of the pre-processing quality check. We can see that the “*dataset\_off\_amazon\_enrichment.txt*” has enough nodes and edges with a reasonable reconstruction runtime. The clustering coefficient difference and the increment of the average degrees can be explained by the removal of non-informative nodes.

Table 1

Quality of the graph samplings

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dataset** | **Nodes** | **Edges** | **Runtime** | **Clustering Coefficient** | **Average Degree** |
| *amazon-meta.txt* | 719'150 | 1'788'712 | 0:20.946405 | 0.174 | 4.974 |
| *dataset\_off\_amazon\_enrichment.txt* | 189'825 | 613'464 | 0:03.691040 | 0.333 | 6.463 |
| *dataset\_off\_amazon\_big.txt* | 126'981 | 527'243 | 0:02.314533 | 0.495 | 8.304 |
| *dataset\_off\_amazon\_small.txt* | 60'707 | 260'472 | 0:01.220927 | 0.562 | 8.581 |

1. Graph exploration and visualization

With the implementation of the features for exploration and visualization, we aimed to a simplified and clear view of the properties of the graph and its communities of nodes.

In the graph exploration, we checked the homogeneity and connection quality of the graph. These values change deeply depending on the pre-processing choice and the size of the sample. We compared the exploration algorithm with our graph and used the Depth-first search (DFS) which was the most efficient one to check if all nodes were reachable. Then, we ran A\* search to compute the smallest path between two nodes and check the minimum size of their leap.

For the visualization module, we managed to display the whole graph with every community highlighted in different color and each community-central node in gold. With the zoom tool of the plot, it is simple to see the clusters. We also made possible to print the members belonging to a specific community. Finally, we retrieved and plotted the degree distribution of the whole graph.

1. Data persistence

We have implemented an online persistence graph on Neo4j. The goal of this database is, by storing all the nodes and edges, to launch fast queries for visualization purposes. We programmed four queries: display of the whole graph, selection of a subgraph (1’000 nodes), display of a desired community (by indicating its index), convert the communities in hypernodes.

1. Data enrichment

We found a newer and more complete dataset of Amazon products online, so we analyzed it to extract common ASINs. We checked for every sample of this new dataset the effect of the merging with our current graph, by executing a community detection algorithm and computing the silhouette index. The merged graph giving the best result has been exported as the enriched dataset, which we used in some of the following analyses.

1. Community detection

To determine which library-ready algorithm of community detection was the most efficient, we tested three group-based on our graphs. We used as a benchmark, the comparison of the runtime, the number of communities and the silhouette index.

Table 2

Comparison of community detection algorithms

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Dataset** | **Algorithm** | **Runtime** | **Silhouette Index** | **Communities** |
| dataset\_off\_amazon\_test.txt | Louvain | 0:00.89 | 0.18 | 1580 |
| dataset\_off\_amazon\_test.txt | Girvan Newman | > 2 days | - | - |
| dataset\_off\_amazon\_test.txt | Modularity | 0:01.38 | 0.26 | 1113 |

As table 2 shows, modularity has a better silhouette index but is 35.51% times slower than Louvain.

Thus, the Louvain algorithm was the most suited for our wide graphs and limited computational power, so we decided to code it by hand.

Our first homemade version of the Louvain algorithm provided comparable results to the official one from NetworkX, but the runtime was very long. To fix this problem, we searched online papers for performance-improving strategies. We found an algorithm based on nodes weight, implemented on top of Louvain main idea. From here we developed a second homemade version of the community detection, designed for directed graphs without weight adaptation.

6.1 Evaluation of the community detection quality

The homemade amazon community detection was running well, but we needed to evaluate its quality. To do so without ground truth, we implemented metrics to compare the quality of our community detection with the library of NetworkX Louvain community detection.

We first compared the silhouette index and the number of communities of the library with homemade implementations. Then, we computed the accuracy, precision, recall and the Jaccard similarity between the homemade implementation communities and the library communities transformed as labels. In order to have a thorough view, we also compared the results with different datasets with runtimes.

1. Results

7.1 Technical results

Table 3 summarizes the results of the community detection of the project.

Table 3

Results of the community detections

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Algorithms** | **Datasets** | **Runtime** | **Silhouette Idx** | **Accuracy** | **Precision** | **Recall** | **Jaccard** | **Communities** |
| **Simple Detection**  **(homemade)** | *amazon\_*  *test* | 0:40:15.93 | 0.21795 | 32.9% | 97.56% | 32.29% | 0.299 | 3’922 |
| **Simple Detection**  **(homemade)** | *amazon\_*  *enrichment* | > 2 days | - | - | - | - | - | - |
| **Amazon Detection**  **(homemade)** | *amazon\_*  *big* | 0:00:12.8 | 0.05008 | 97.01% | 95.36% | 97.01% | 0.925 | 21’157 |
| **Amazon Detection**  **(homemade)** | *amazon\_ enrichment* | 0:00:17.5 | 0.19851 | 97.54% | 89.76% | 97.54% | 0.875 | 17’818 |
| **NetworkX**  **(Louvain)** | *amazon*  *\_big* | 0:00:21.4 | 0.02363 | - | - | - | - | 23’521 |
| **NetworkX**  **(Louvain)** | *amazon\_*  *enrichment* | 0:00:17.7 | 0.08338 | - | - | - | - | 26’887 |

First, we can see that the simple (first version) and the improved (second version) homemade detections demonstrate comparable community results, except for the runtime, which is excessively high on the first one. In fact, we even needed to use the smaller sample *dataset\_off\_amazon\_test*, because with the big dataset we could not get a result in a reasonable amount of time. This proves the improved version actually is more efficient. Furthermore, it has a better runtime and silhouette index than the NetworkX Louvain algorithm (+0.02645 on the *dataset\_off\_amazon\_big.txt* and +0.11513 on the *dataset\_off\_amazon\_enrichment.txt*). This result also proves that the enriched dataset is more effective than the original sampled one. Furthermore, the scores of accuracy, precision and recall demonstrate a close relationship between the community detected on the NetworkX and the improved homemade algorithm, proving that the logic of Louvain is maintained despite in our own implementation. Finally, the number of communities is lower with our (second) version, but still broad enough to extract meaningful popular nodes.

7.2 Relating to the hypotheses

To answer the main hypothesis of this project, we computed a correlation analysis between the centrality scores of the nodes categorized as community-populars and their sales rank. The Spearman’s test resulted in a correlation of -0.15 (*p* < 0.01). This result shows that there is a negative relationship between the selling success of a product and its centrality in the network, which reverses what were our expectations. We therefore tested the correlation between the 15 most popular products only, to investigate if the majority of popular products do not relate to their success of sale, which could shape negatively the overall correlation. We got a Spearman’s score of -0.268, but it was not statistically significant (*p =* 0.334).

Table 4

Five most popular products of the dataset and their salesrank.

|  |  |  |
| --- | --- | --- |
| ASIN | Betweenness Centralities | Salesrank |
| 890964653 | 0.00976 | 205,435 |
| 486411214 | 0.00777 | 3,212 |
| 7894799433 | 0.00641 | 27,441 |
| 486250237 | 0.00585 | 38,536 |
| 764585975 | 0.00585 | 15,356 |

1. Conclusion

To conclude, we were able to implement an effective community detection homemade algorithm even with an initial dataset not well suited for community detection, since the silhouette index score suggested a tendency of small distance between clusters and a high distance between nodes of the same community.

Furthermore, we found a small negative correlation, meaning that the sales success of a product is inversely proportional to its centrality in the graph. One explanation could be that new trendy products are temporarily in a higher rank than popular classic ones that remain steady popular products on the long run, such as the Bible.

A strength of our application lies in having efficiently reduced the dataset size by focusing on few features, which made us gain computational power. The limitation of our application is the original low cluster quality, which we could improve only to a certain amount, and the still long runtime needed for of some visualization plots.

A direction of improvement implementation could lie in connecting ASINs with different criteria, then with the similars (like we did). Moreover, a correlation analysis using more recent data to validate our hypothesis could be performed.

To summarize, we believe that by our analysis we got to gain valuable insight of the graph properties of products networks, which has the potential to help business intelligence decisions.