UCLouvain

LINMA2472 - Algorithms in Data Science

Network Analysis on Harry Potter and the Sorcerer's Stone, a novel by J. K. Rowling

Team Member Names:
Sarra Laksaci - sarra.laksaci@student.uclouvain.be
Mojgan Giahi - mojgan.giahi@student.uclouvain.be
Sixto Castro Redrobán - sixto.castro@student.uclouvain.be

Teachers:
Jean-Charles Delvenne
Gautier Krings
Course Assistants:
Rémi Delogne
Loïc van Hoorebeeck
Nizar Bousselmi

A report submitted in partial fulfilment of the requirements of
the UCLouvain for the degree of
Master of Science in *Computer Science* and *Data Science*

October 20, 2021

Abstract

Harry Potter and the Philosopher's Stone is the first novel in the immensely popular Harry Potter series by British writer J.K. Rowling. The author was praised for creating well-rounded characters and a fully realized wizard universe that coexisted with the present world. Harry Potter and The Philosopher's Stone is a story that revolves around the legend of the Philosopher's Stone, a stone that can transform any metal into gold and bestow immortality. Harry Potter, the main protagonist, is able to overcome the hard difficulties that he faces and understand his real nature, which is the world of magic. Also, the power of love is prevalent throughout the entire novel.For all these reasons we can see an opportunity of modeling this massive data through networks. In this project, we pay a closer look at social networks, which are a subcategory of this structure that focuses on representing a relation between entities, thus allowing us to study the interactions of the given entities in the defined world of our network. More precisely, we will be working on the book mentioned: Harry Potter and the Philosopher's stone, from which we will extract a co-occurrence network of characters modeled as an undirected weighted graph where the nodes represent the characters of the story, and the edges describe the relation of appearance in the same paragraph. In the second part of this document, we will analyze the properties of the inferred graph and proceed to further studies, including community detection, k-core decomposition, and preferential attachment network generation. Finally, we will focus on the task of maximizing the spread of information through a social network by solving the influence maximization problem using a greedy algorithm: Greedy Hill Climbing Heuristic, followed by a comparison study of ICM results.

1 Preprocessing: Characters and Network Creation

Harry Potter and the Philosopher's Stone is rich in character development and has a wide variety of characters. Most of the characters in this book are from the world of magic. Characters range from students and teachers at Hogwarts to animals and other magical beings. Moreover, this book contains a significant number of interactions between the characters. Therefore, in our opinion, the novel could stand well for studying the characters and accomplishing our project.
This section focuses on the steps followed to build the network as well as the challenges encountered while doing so. The overall process includes gathering the extensive list of characters and splitting the text format into clean sections and finally creating the data structure which represents the network. We regrouped a total of 120 characters across 17 chapters for 1888 paragraphs.

1.1 Challenges

- The characters list is a prime tool used to gather the counts of appearance of each character. However the different identifications used for a single character can skew the results, for example the protagonist of the story: Harry Potter can be referred to as Harry, Potter, or The Boy Who Lived. Another example would be the antagonist Lord Voldemort who's real name is Tom Marvolo Riddle but mentioned in the book as You-Know-Who.
- The clean sections of text should represent the separated paragraphs. The first intuition would be to follow the linguistic separation which is represented as indentations and new lines.However this process isolates the speeches, thus cutting down the information of a relation between the speaker and the replier (since the speech and the response are considered as paragraphs)

1.2 Network Creation

In this section, we will briefly go through the details regarding the network creation and character extraction process.

- **Building the characters structure:** The initial structure is based on an online source [1] which lists the different characters of the story in natural language with additional comments. To formalize this representation we used a dictionary where the key is an integer and the value is a list of possible names to refer to the given character. This list includes the full name (found in online ressource), first and last name separately (using split on white space when it does not produce noise. ex: Hermione Granger -¿ Hermione, Granger. but Milk Man cannot be split),surname (added manually after research for the few main characters),

- **Splitting the text into clean sections:** The book was initially split according to new lines, as this defines the majority of the paragraphs in the text file after stripping the chapters titles. This gives us a list of string entities where each element is a separated paragraph. To solve the challenge mentioned earlier regarding speech, we decided to merge all sentences including quotation into the previous element (paragraph) in the list. This approach allows us to collect whole conversations together with their context. The end result is a list of the clean sections, consisting in 1888 paragraphs..

- **Forming the adjacency matrix:** The adjacency matrix is a representation of a weighted graph where element A[u,v]= x means that node u and node v are linked by an edge of weight x. In the case of this project, this means that characters u and v appear together in x paragraphs of the book. In order to form the adjacency matrix we parsed the content of the clean sections list extracted from the book. Using the characters dictionary, we create an appearance list for each section (the list is cleaned after each section) that includes the key of all characters mentioned in the viewed paragraph. For each couple of values in the appearance list (u,v), we increase the value of A[u,v].

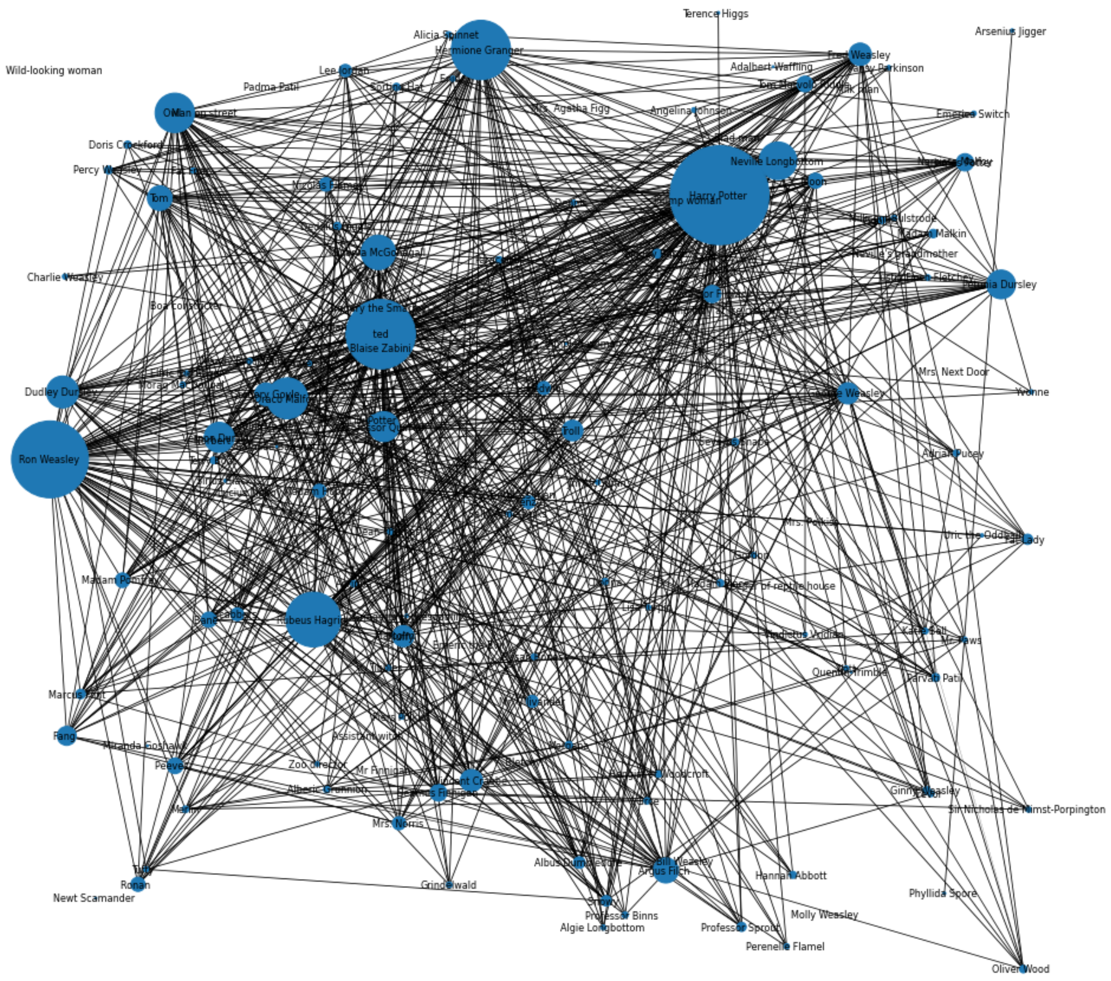

Figure 1: Co-occurrence network of characters in the

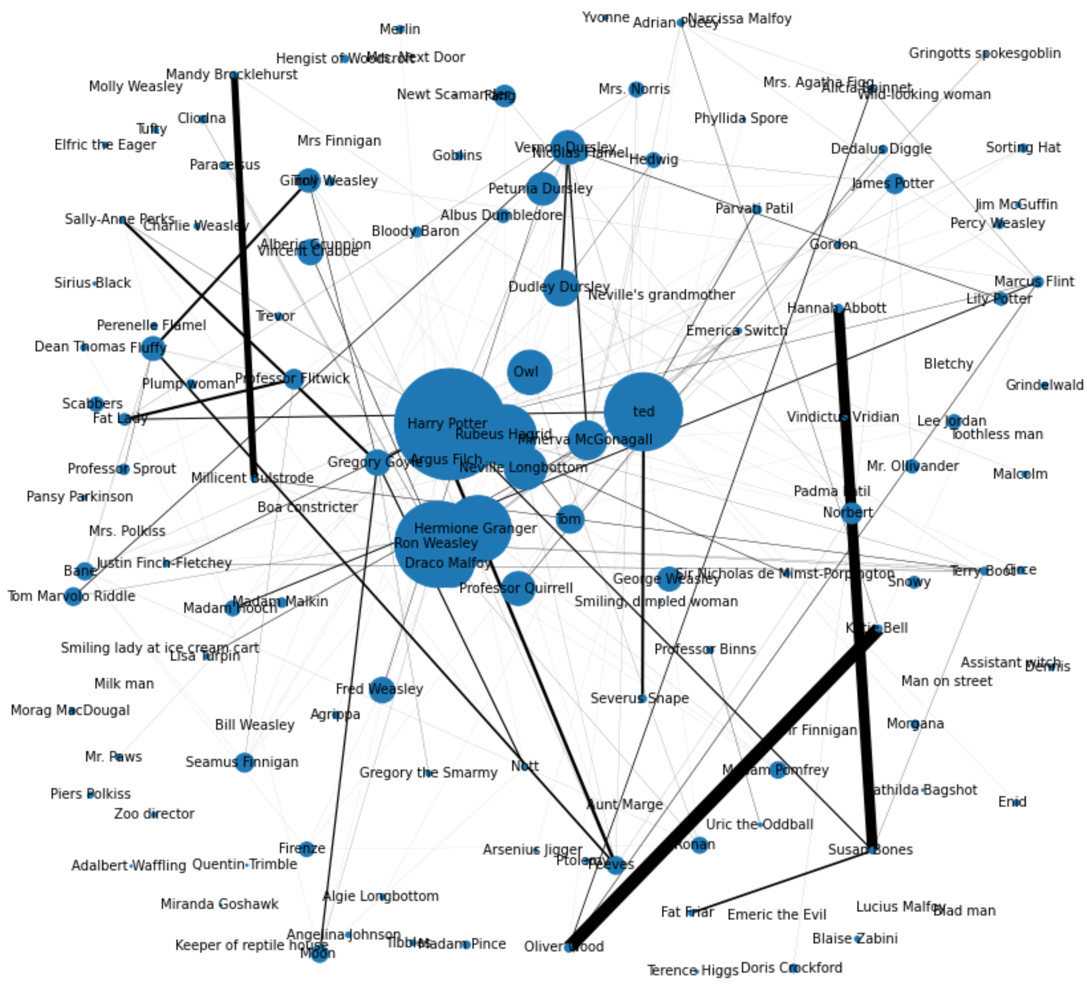

Figure 2: Co-occurrence network of characters with edge width variation based on weight

2 Network Properties

In this section, we will review the network characteristics such as Degree Distribution, Clustering Coefficient, Community Structure, and so on.

2.1 Degree

The network includes 120 characters. Its average degree is 10.88. At first glance, it can be seen in Figure 3 that the distribution does not exhibit an excellent right-skewed distribution, as observed in the many real-world networks. The Top five characters with the highest degree in the network are respectively "Harry Potter", "Ron Weasley", "Hermione Granger", "Minerva McFinagall", and "Rebeus Hagrid". All these names are major characters so that the absence of each of them can change the whole story. For instance, Harry Potter is the main character, or Ron Weasley and Hermione Granger are Harry Potter's best friends and accompany Harry almost everywhere, Minerva McFinagall is the head of Gryffindor house and and deputy headmistress of hogwarts, and also Rebeus Hagrid who loves harry and helped him many times in the story. As it's clear, all these characters have a tied connection with the main character, so it is expected that the author more frequently mentioned their names in the paragraphs. The degree assortativity coefficient for our network is -0.32. This negative value indicates that the network is degree disassortative, meaning that high-degree characters in the network associate preferentially with low-degree characters. We can tell it from the story where the name of the main characters appeared with name of the ordinary characters.

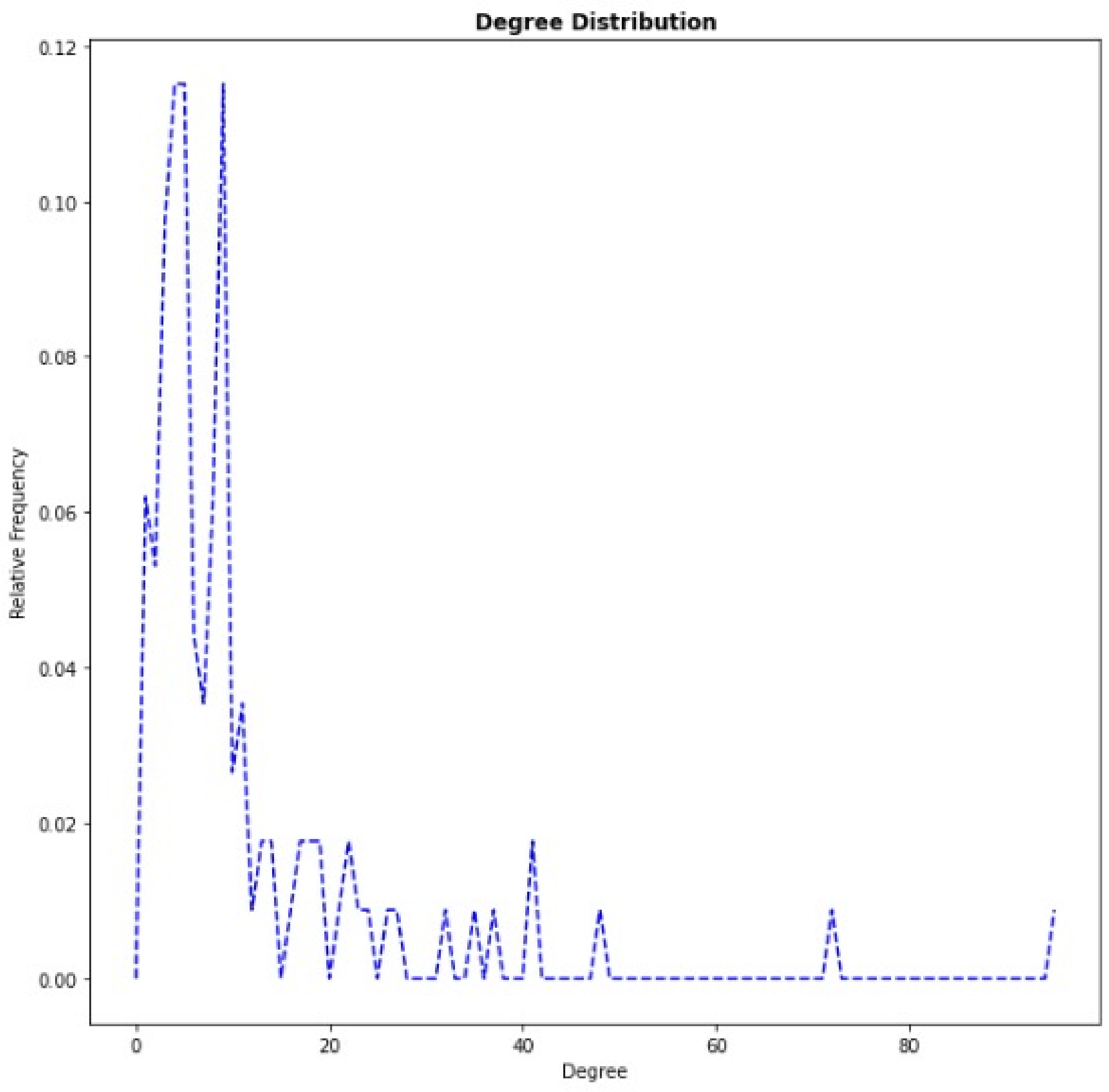

Figure 3: Degree Distribution of The Network

2.2 Clustering Coefficient

The assortativity metric is the Pearson coeficient of degree between 2 linked nodes. It is said when the correlation is positive, the graph is assortative, that is, nodes of similar degree are connected (For example: nodes with high degree are connected with other ones with similar degree).
In our scenario, the coeficient is negative (degree assortativity coefficient: -0.368), therefore our network is disassortative, that is, the nodes with high degree are connected with the ones with lower degree (or viceversa) and in our case: Harry potter is well connected to many characters but these ones don't have as much connections as Harry, in other words they have a low degree level.

2.3 Louvain Algorithm

The Louvain algorithm is a method for detecting communities that tries to find the best partition for a graph. And the value to optimize is the modularity which represents a value defined between $[-0.5, 1]$ where negative values (-0.5) indicates non-modular clustering and positive values refers to modular clustering $(1 \rightarrow full)$.

This algorithm algorithm functions as follows:

- First, each node is represented as a community (N nodes and N communities). Then every node will look around its neighbours and will be removed from its own community to join the community of the neighbour which produces the maximum increase of modularity. This phase 1 is repeated until finding a local maximum modularity where each node can be considered several times.
- For the phase 2, once the local maximum is achieved, a new network is built where each nodes represents a community from the previous phase, Any link belonging to the same community is represented as a self-loop, and the weight of links between different nodes (communities) is the total weight of the links between nodes of these communities.
- The 2 steps are repeated several times which produce a hierarchical network decomposition.

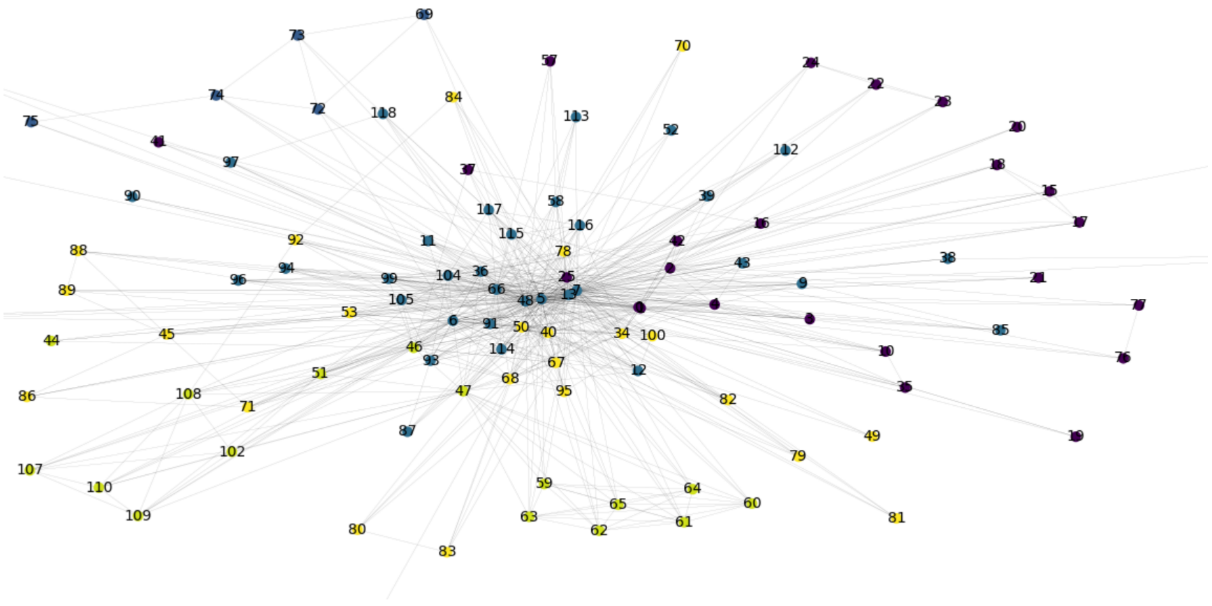

Figure 4: Detection of communities: Louvain Algorithm

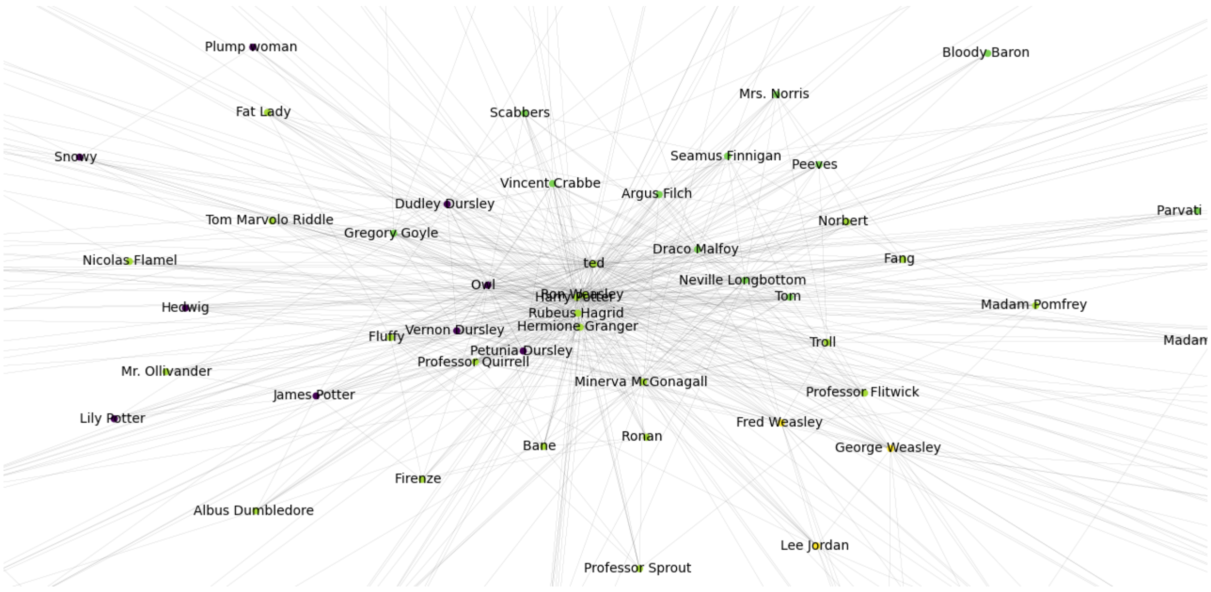

Figure 5: Detection of communities: Louvain Algorithm (Character names)

In Figure 4, we can observe 3 core communities in the graph thanks to the best partition calculated by the Louvain Algorithm for their detection, and each node is represented by its own ID (character) and respective links. In Figure 5, for better understanding about the clustering of the characters, a graph was plotted (zoomed in) and each node contains the name of the character. Let's analyse its distribution: The green community represents the principal characters and Harry Potter is the center of all these nodes, and they are people that attended classes in Hogwartz such as: students or teachers; while the yellow cluster represents people from sports team (Quidditch game: player, commentator) and we see names like: Fred, George, Lee Jordan, Marcus, Oliver, Alicia, Morgana, etc. And the third community is the purple cluster that presents secondary characters that were partially present as Lilly and James Potter, and others are animals.
The modularity of the graph is 0.119 which represents a positive modularity and tells us that a modular clustering exists in our graph and for that reason we can identify in a good way the distinct clusters (the 3 ones mentioned above).

3 K-Core Decomposition

3.1 Explanation

A k-core decomposition of a graph is a sub-graph in which every node has a degree of at least k, which could be any integer. For instance, the 0-core of any graph is always the graph itself. The degeneracy of a graph is the smallest value of k for which it is k-degenerate. The maximum core value that can exist in the graph is also known as graph degeneracy. It means that it is the largest K we could decompose the graph into. We can utilize it for identifying the clusters of a graph.

3.2 Implementation

The challenge here was to implement it without using the built-in function of NetworkX. Instead. we have created a for loop which goes through all nodes and checks if there is a node lower than K, and if so, it will remove that node. However, based on our investigation, it can only be used if all nodes are sorted. The reason is that the degree of a node might be lower than K after the algorithm finished checking, and so that specific node will not be removed. That is the reason why we have implemented a sorting function that sorts the nodes by their degrees and then applies the algorithm to prevent this inconsistency.

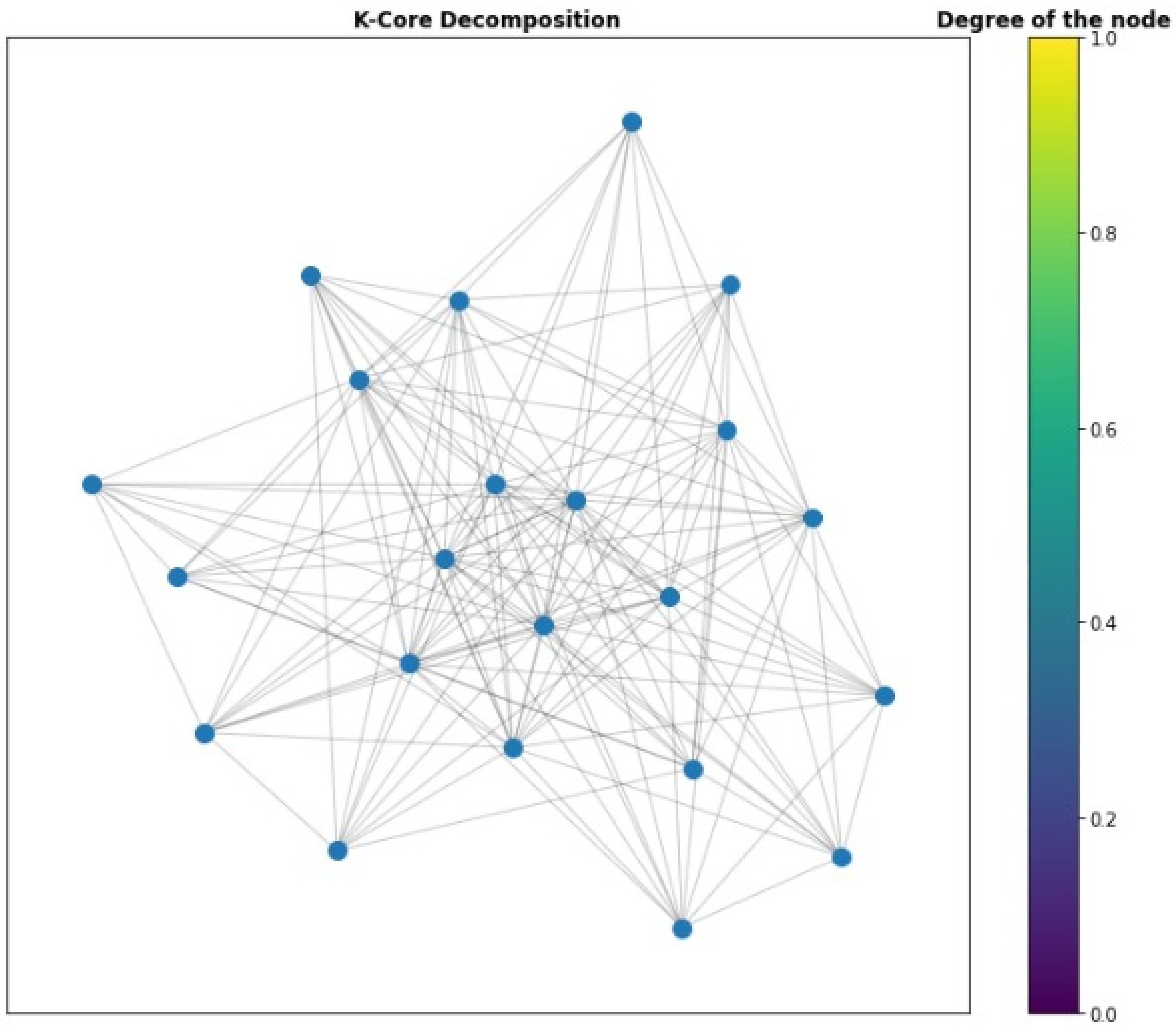

Figure 6: K-Core Decomposition (K=9)

3.3 Result

We considered $K = 9$ and our Graph has a high mean degree (Max 11) and we have 22 nodes for it. For maximum K we have 3 nodes. There were no unlinked subgraphs for any of the k-core subgraphs, which is likely to indicate that there are no clusters

4 Preferential Attachment (Barabási-Albert) Network Analysis

4.1 Explanation

Briefly, the Barabási-Albert is an algorithm that is being utilized to generate random scale-free networks using two simple mechanisms, growth and preferential attachment, which are responsible for the emergence of scale-free networks. To be more specific, the possibility that a new node can be attached to a particular node is proportional to that specific node's degree. In other words, as higher as the node's degree is, the higher are the chances to that node to be connected to a new node. Hence, older nodes are more likely to achieve higher degrees than the following nodes. As long as the network total degrees increases, the probability of new nodes establishing further connections reduces. Intuitively, the preferential attachment can be understood if we think in terms of social networks connecting people.

4.2 Implementation

We utilized Barabási-Albert Graph function to generate our network. This function is based on two variables: the final node number of our graph (N); and the number of edges that each new node will connect to the pre-existing network (M). We opted for a fixed number of edges of 10 to attain an average network degree of 16.72. It means the newborn nodes will connect to 10 pre-existing nodes.

4.3 Result

In regards to the Assortativity Coefficient, the Barabási-Albert model presented a value of -0.12 while the book's network -0.32. The negative value means that high-degree nodes associate preferentially with low-degree nodes in both networks affirming a dissortativity characteristic in both networks. The algorithm divided the book network into 7 communities with a modularity value of 0.14.

5 Rumor Spreading

Diffusion on a social network is the task at hand if we want to spread a rumour across a population. Moreover if the goal is to inform the maximum number of people in the minimum time, this situation describes the Influence Maximization Problem. In a formal way, the influence maximization problem is for a given k finding the set of nodes of size k with maximal influence. Where the influence of a set of node A is noted a $\sigma(A)$ =number of active nodes at the end of the diffusion process starting from initial set A of activated nodes.

5.1 Greedy Hill-climbing heuristic

Algorithm

```
Given a graph G(V,E) and an ICM model defined on G
1) Generate a large number of realisations $X_i$ of the ICM model defined on G
2) Initialise $A_0$=[]
3) while length($A_0$) <k: Find v in V/$A_0$ that maximizes $\sum P(X_i)\sigma_{xi}(A_0$ U v) Add v to $A_0$
```

Implementation

- Set Size K : the size of initial set A_0 is equal to 0.5% of total nodes thus k=6
- ICM simulations : Since the heuristic relies on partial randomness, flipping a coin to activate edges and assigning P(x) to each realisation, we reiterate the operation over a number of simulations. We set this hyper-parameter to 10
- Graph Percolation in ICM : we simulate the ICM by proceeding to a random percolation of the graph. An edge is activated or not according to the result of a flip of a coin with a probability p. All probability distributions over edges in a simulation X are stored in list X. The new representation of the graph is a matrix where $A[i, j] = 1$ for an active edge between node i and j, or 0 else
- Influence of a node set : we introduced $\sigma(A)$ as the number of active nodes after diffusion. This number is calculated such as each node linked to an initial node of A_0 via an active edge is counted. We implemented this

Results

test n°	nb simulations	Initial set A
1	10	Harry, Ron, Hermione,Mr. Dursley, Mrs. Dursley,Pr Binns
2	10	Harry, Ron, Draco Malfoy,Ronan, Hagrid, Hermione
3	100	Pr Binns, Hermione, Harry, Mr. Ollivander, Ron, Grindelwald
4	100	Harry, Ron, Hermione, Bane, Ronan, Hagrid

Table 1: Results of Greedy Hill Climbing Algorithm with k=6

5.2 Independent Cascade Model

ICM: Independent Cascade Model is a model for diffusion on social networks that works as follows: given a graph G(V,E), at every step if a node u is activated it only gets one chance of infecting its neighboring nodes with a probability of succeeding $p_{u,v}$. The last step is when no more nodes can be infected.

Implementation

- in this application we consider an ICM as the generation of a random number on each edge (as in the implementation of the greedy algorithm)
- As the Influence Maximization Problem is NP-Hard for the ICM, we will be conducting a comparison of ICM results starting from intial set of size k with highest degree nodes and initial set of size k with random nodes.

Results

- Highest degree nodes(A1): Harry, Ron, Hermione, Draco, Harged, Mcgonagall
- Random nodes (A2): Owl, Dudley, Neville Longbottom, Padma Patil, Olver wood, Fred Weasley
- We notice that the application of ICM with both sets A1 on the same realisation X of probabilities results in a better influence for set A1 of highest degree. this can be explained by the fact that higher degree nodes have more chances of infecting neighbors than lower degree nodes