NUI MAYNOOTH

Ollscoil na hÉireann Má Nuad

Online Social Networks: Measurements, Analysis and Solutions for Mining Challenges

by

Rana Maher

Master of Science (MSc.) Thesis

Hamilton Institute
National University of Ireland Maynooth
Maynooth
Co. Kildare

October, 2017

Research Supervisor: Dr. David Malone

Head of Department: Prof. Ken Duffy

Declaration

I hereby certify that this dissertation is the result of my own work that has not been taken from work of others, does not violate any law of copyright and includes nothing which is the outcome of work done in collaboration except where specifically mentioned in the text.

Signed by: Rana Maher

Date: October, 2017

Abstract

In the last decade, online social networks showed enormous growth. With the rise of these networks and the consequent availability of wealth social network data, Social Network Analysis (SNA) led researchers to get the opportunity to access, analyse and mine the social behaviour of millions of people, explore the way they communicate and exchange information.

Despite the growing interest in analysing social networks, there are some challenges and implications accompanying the analysis and mining of these networks. For example, dealing with large-scale and evolving networks is not yet an easy task and still requires a new mining solution. In addition, finding communities within these networks is a challenging task and could open opportunities to see how people behave in groups on a large scale. Also, the challenge of validating and optimizing communities without knowing in advance the structure of the network due to the lack of ground truth is yet another challenging barrier for validating the meaningfulness of the resulting communities.

In this thesis, we started by providing an overview of the necessary background and key concepts required in the area of social networks analysis. Our main focus is to provide solutions to tackle the key challenges in this area. For doing so, first, we introduce a predictive technique to help in the prediction of the execution time of the analysis tasks for evolving networks through employing predictive modeling techniques to the problem of evolving and large-scale networks. Second, we study the performance of existing community detection approaches to derive high quality community structure using a real email network through analysing the exchange of emails and exploring community dynamics. The aim is to study the community behavioral patterns and evaluate their quality within an actual network. Finally, we propose an ensemble technique for deriving communities using a rich internal enterprise real network in IBM that reflects real collaborations and communications between employees. The technique aims to improve the community detection process through the fusion of different algorithms.

Acknowledgment

The work in this thesis could not exist without the help and support of many people during Master Thesis research project. I am hugely indebted to friends, family and colleagues.

I would like to express my sincere gratitude to my supervisor Dr. David Malone for his always guidance to provide me with the necessary fundamental to fulfill my research project and for the many hours spent in discussing to point me to many interesting problems in order to pursue my research in the right direction. I am also grateful to everyone in Hamilton Institute and all the people working there. In particular the administrative stuff in the institute, Rosemary Hunt and Kate Moriarty who always facilitate any administrative work needed to accomplish my research.

Also, I would like to thank all the researchers and scientists I have referred to in my thesis. I spent many hours reading and studying their resources. They provided me with rich theories and very useful insights about the field. I am very grateful to the people were involved in this project for their continuous support and cooperation: Marie Wallace and Faisal Ghaffar from IBM Software Lab.

I would not have accomplished this road without the continuous support, affection and love showered by my parents Maher Elgendy and Amany Dougheim. Their patience, understanding and continuous encouragement are greatly appreciated. My heart felt regard goes to my father in law Assem, mother in law Azza for their moral support and endless love. Thanks to my sister Dalia, my brother Karim for their selfless love and care which contributed a lot for the completion of my thesis. I consider myself a lucky person to have such a lovely and caring family, standing and supporting me unconditionally.

Last but not least, I owe many thanks to a very special person to my heart, my beloved husband, Haytham Assem. He is the person behined all the scenes who made the completion of this thesis possible. During these years, he has been a constant source of support and ultimate motivation. He was always around at all times and helped me to keep things in perspective. I deeply appreciate his belief in me and the huge support he showed to help me achieve this degree. Thank you my great husband for giving me the key to success and being my source of inspiration in life. Words would never express how I am truly thankful for having you in my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, we present the motivations behind the presented work and our contribution in this thesis as well as proposing an overview of the material and the scope in the next chapters.

1.1 Motivation of the thesis

Online social networks have shown a huge growth in the last decade. With this enormous rise of the online social networks like Facebook[1], Twitter[2] and LinkedIn[3], researchers have gained the opportunity to have access to online social networks data and to track the behavior of people and their interactions. These online networks started with only hundreds of people but with the drastic growth of such networks nowadays, this sheds the light for researchers to gather deeper insights about people and see how they behave in the network. They also provide information about communities, roles and user actions. Social graphs are popular structures for modeling relationships, interactions and communication between users, organizations or even groups. They are the commonly used way of modeling the shared information and ideas over the Internet [2]. Network analysis is a collection of techniques to calculate various metrics for a social graph.

There is a positive look on the field of mining, analyzing social networks and their vast applications in terms of research and impact on business. But still the area is not that mature and there are a lot of challenges which require novel solutions or techniques from different disciplines. Some of these challenges are technical challenges and others are social and human challenges.

An example of technical challenges is the networks dynamics, as most of the networks turned from a static to dynamic due to the evolving nature of these networks [3]. Being able to deal with this evolution is becoming crucial and a lot of research need to be done to create models to deal and understand this growing of time-stamped networks

[1] https://www.facebook.com
[2] https://twitter.com
[3] https://www.linkedin.com

either for research or business goals. Also, data preparation can create technical challenges; it requires developing techniques to facilitate managing, cleaning, documenting and anonymizing data. Other challenges include evaluation and the difficulty of having a reference metric either due to the difficulty of collecting and sharing data or due to the difficulty of formulating a ground truth even if the data is available. Hence, the generalization of results of one performed experiment became a challenging task.

As for the social and human challenges, these include privacy problems, which requires defining the right balance whether if the information is related to the situation, individual or organization. Other ethical and legal issues place limitations on making use of the data even in research purposes. In addition to the aforementioned challenges, defining the community structure involves defining the appropriate level of communities and subcommunities that need to be targeted and how the existing links can be interpreted.

In this thesis, we consider some of the challenges in social network analysis like networks dynamics, community structure, evaluation, etc. We contribute and provide some techniques, methods and frameworks to deal with these challenges, this will be discussed in detail in the following sections.

1.2 Contribution of the thesis

The main contribution of the thesis is to provide and introduce some solutions for the analysis and mining challenges in social networks which may remove some of the obstacles that obstructs the analysis process. First, we have provided a prediction approach that can be used to predict the execution time and the performance of the analysis specifically to deal with evolution of networks phenomena and solve the challenge of dealing with huge networks. This also helps in defining the required hardware and reflects an estimate about the expected performance before performing the analysis. It is argued that this approach could be used further across any network size or any type of hardware. Second, we present an analysis that provides insights to better understand the ground truth communities within the Enron [4] email network through analysing the network topology and evaluating the performance of different community detection approaches. The derived insights can help in providing insights on the communities within the Enron network without knowing the real ground truth and also can help in recommending the desired network representation. Finally, we focus in this thesis on optimizing the quality of the derived communities in the network by developing an ensemble community detection approach and proposing an approach to derive hybrid clusters that is argued in resulting better and more optimized communities. The approach can be applied on different types of networks and supports ensemble of communities, either from different approaches or from different network representation of the same network.

1.3 Structure of the thesis

The thesis is organized as follows: Chapter 2 gives an overview of graph theory and the mathematical background that is required for core contribution chapters. Chapter 3 provides an overview of the concepts and the techniques for analysing graphs. Also, it presents an overview on the state-of-the-art work and highlights the challenges that can obstruct the network analysis process. Chapter 4 presents our contribution regarding dealing with the evolution of networks and predicting the performance in advance. Chapter 5 and 6 provide our techniques for how to deal with the community detection problem, the lack of ground truth for social networks and our optimization approach for deriving better communities respectively. Finally, Chapter 7 concludes the thesis highlighting potential areas for future work. This structure can be described in detail as follows:

Chapter 2 introduces an overview of the mathematical concepts of graph theory which provide the basic concepts and applications used for network analytics. Several definitions and theorems are reviewed to give a theoretical overview of the hidden mathematics that provides the way to analyse a social network using social graphs.

Chapter 3 aims to present background of the social network analysis area and its fundamentals. Different analysis techniques and methods are proposed. A literature review provides the commonly used applications and the work done in this area. Finally, a categorization for the main challenges that have gained the attention of many researchers in this field is presented.

Chapter 4 introduces an approach based on predictive modeling that can tackle one of the most popular challenges in social networks field, which is the dynamic nature of the real-world social networks that dramatically change over time. Incorporating features of dynamic or evolving networks requires repetitive computations and so performance of the calculations is essential. Thus, we introduce a performance prediction approach that can be used as a tool to provide an approximate execution time needed to analyse a given network. The approach is based on applying supervised machine learning models by training the performance of calculating some popular network measures with different mining tools. Our approach aims to predict the execution time for a given network before even starting the analysis process. Predicting the performance can help in choosing an appropriate mining tool that can suit the size of the given network and can assist in deciding if there is a need for additional hardware (e.g. High performance computing clusters). The work in this chapter addresses the fertile area of how to deal efficiently with the evolving networks.

Chapter 5 addresses the implications of discovering communities while lacking ground truth. We present in this chapter our approach, which measures the topological properties (e.g. size distribution, density) within the communities and assess the quality of the derived communities in advance in order to recommend the most appropriate communities without the need of a ground truth for reference or evaluation. We performed this detailed analysis on the Enron email network dataset to study the topology of the derived communities. We conducted this study on different representations of this network arising from different email interactions. We deal with each interaction as a standalone network. Then, we evaluate the results through comparing the derived communities, which result from the interplay of both different community detection approaches and the different interactions using commonly used evaluation metrics in the field. We provide some heuristics, insights and recommendations on how to select the appropriate approach and what is the most informative interaction representation for the Enron email network. The objective of this contribution is to try to find a closer view of real world communities in the network that can act as a ground truth especially when the real structure of community is hidden.

Chapter 6 addresses the criticality to establish the best methodologies for community detection of unknown structure in networks. We studied the challenges for optimizing communities through proposing and implementing an ensemble/hybrid fused approach for identifying complex and mixed structures of nodes and links communities, called **Multi-criteria Community Fusion (MCF)**. The main contribution is the development of the MCF approach that accommodates hybrid communities for any type of networks and supports the various existing community detection techniques. Our ensemble approach and experiments were performed on a rich real internal enterprise network within IBM called *IBM Connections* and not on an artificial network. Thereby, our results reflect the real nature of networks. This enterprise network provides information about collaborations, interactions, shared information and personalized relations that took place between employees. The proposed approach performs well when applied to certain types of subnetworks. Each subnetwork reflects certain relations and interactions between IBM employees. Our results conclude that not every ensemble approach should perform better than single methods as expected and that the structure of the network plays a recognizable role in the quality of the results.

Chapter 7 highlights the conclusions of the achievements and the contribution of the thesis. It also suggests the possible future work that can be further researched in the area of social network analysis.

1.4 Publications

The work and the contribution in this thesis has been accepted and published in the below international conferences. My contribution through these publications lead to Chapter 4, Chapter 5 and Chapter 6 respectively.

1. *Rana Maher and David Malone. "Analysing and Predicting the runtime of Social Graphs." Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom)(BDCloud-SocialCom-SustainCom), 2016 IEEE International Conferences on. IEEE, 2016, Atlanta, USA.*

2. *Rana Maher, David Malone and Marie Wallace. "The Impact of Interaction and Algorithm Choice on Identified Communities". Social Media, Wearable And Web Analytics (Social Media), 2017 International Conference On. IEEE, 2017, London, UK.*

Chapter 2

The Power of Graph Theory

The purpose of this chapter to touch lightly on the topic of graph theory and familiarize the reader with the basic concepts, and, consequently, a list of useful definitions and notations. All of these definitions constitute the basic structure of social networks analysis. At first we will provide an overview of the history of graph theory and how the concept of modern graph theory arose. Then we will try to summarize the most relevant graph theoretical concepts, definitions and formulas that are considered the mathematical background for network analytics.

2.1 Graph History

The origin of the theory of graphs started with the problem of the seven Köningsberg bridges as shown in Figure 2.1, when Euler was asked to find a nice path across the seven Köningsberg bridges and this path should cross over each of the seven bridges exactly once [5]. Köningsberg is situated at the sides of the river Pegel which comprises two big islands. Euler thought that the only clue to solve this problem is to derive a sequence for the bridges to cross. Euler mapped the problem into mathematical graph structure. He mapped land masses to *vertices* and the bridges to *edges*. He pointed out that during a walk: If the walk starts from a certain bridge, it should then leave from another one, given the fact that every bridge should be crossed only once. This should mean that the number of bridges that touch the land mass should be even. But, there is a contradiction here as all the land masses were touched by an odd number of edges. Therefore, Euler concluded that this walk exist iff the graph is connected with exactly 0 or 2 vertices having an odd degree. By proving this, Euler put the essential base of modern graph theory.

2.2 Preliminaries

Graph theory is the main study for graphs where simple data structure are mapped into set of vertices and links between these vertices. Sometimes a vertex is called a node and an edge might be called a link. For an overview on the general theories of graphs, the

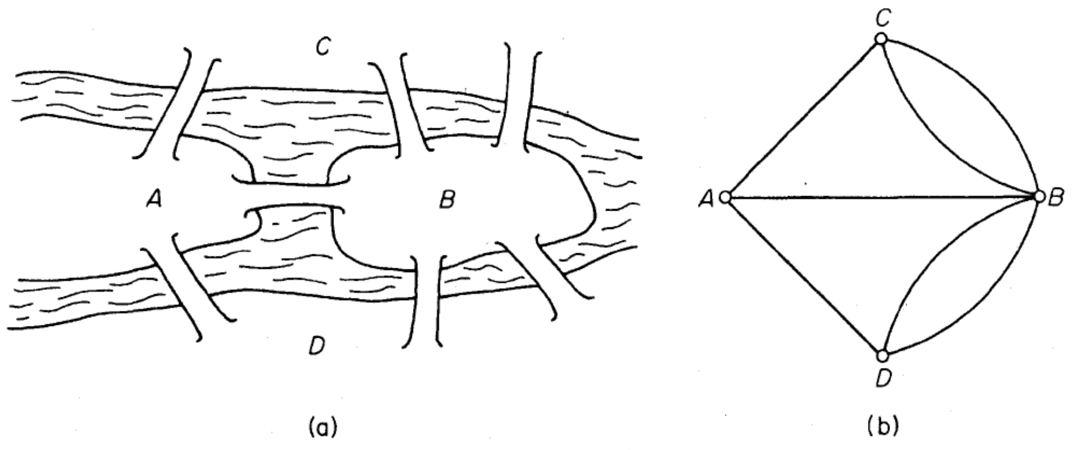


Figure 2.1: The seven bridges of Köningsberg and their graph [1].

reader can refer to [6] and [1]. As mentioned before, some fundamental notation and definitions used throughout this thesis are presented in this chapter. Others will be given later in the next chapter.

We begin by defining a **simple graph** or **graph** G is a pair of sets (V,E) where E is a set of two sets of V. Graphs can be represented by diagrams, however these diagrams are not graphs. Graphs can be weighted, labeled and colored which we will describe later.

While a **subgraph** $H=(T,R)$ of a graph $G=(V,E)$ is a graph where $T \leq V$, $R \leq E$. The subgraph is induced by a set of vertices T is the graph obtained by including the edges between the vertices of T.

2.3 Paths, Cycles and Walks

Finding the optimal solution to measure the shortest path within a graph is indeed not a trivial task due to the difficulty of choosing which vertex should be chosen next at each point of the path. Some of the common measures used to tackle this point include: a **path** which represents a sequence of distinct vertices involved while following a sequence of edges through out the network. A path is named as a **geodesic path** when it represents the shortest path between two vertices having the minimum number of edges. A **walk** is an another measure which represents an ordered sequence of vertices including the repeated ones. On the other hand, a connected sequence of distinct edges in the graph with repeated vertices but no repeated edges is named as a **trail**. However, when a trail starts and ends at the same vertex with no vertex repeated, it forms a **cycle**.

2.4 Graph Properties

The following are some commonly graph properties and characteristics that deal with the number of vertices and the number of edges for any graph G. These properties can give a high level overview about the structure of any graph.

- The **size** is the number of edges E in the graph while the *order* is the number of vertices V in the graph.
- The **path length** L is the distance $d(i, j)$ between any vertex of the graph and every other vertex. This distance corresponds to the number of edges that should be crossed to move from vertex i to vertex j, this is what is called by the **shortest path** that is defined in [7] as: a path length L of a graph is the median of the means of the shortest path lengths connecting each vertex $v \epsilon V(G)$ to all other vertices.
- The **diameter** of the graph G is the maximal distance between any two vertices in the graph. It is also known by the maximum *eccentricity* across all vertices such that the eccentricity of a node i is the maximal shortest path distance between i and any other node. While the *radius* is the minimum eccentricity across all vertices in the graph. According to the computational complexity of measuring the radius and the diameter they are always calculated through measuring the eccentricity of a random of vertices in the network.
- The **degree** of vertex i is the number of incident edges to this vertex. An isolated vertex has degree 0 while the degree of a self loop counts 2.
- The **neighborhood** of a vertex i is the set of all the vertices that are connected to vertex i. The formal definition of neighborhood is:

  The neighborhood $\Gamma(i)$ is the subgraph consisting of all vertices adjacent to i (not including i itself).
- The **clustering coefficient** of the vertex is measuring how the neighborhood subgraph of a vertex i is well connected. It is the fraction of actual edges and possible (expected) edges between the neighborhood of vertex i. While the clustering coefficient of the graph G is the average clustering coefficient across all vertices.

2.5 Graph Characterization

Graphs differ in several ways based on the types of edges that connect the vertices to each other. The focus in this section will be on the characterizations of different types of graphs as shown in Figures 2.2, 2.3, 2.4 and 2.5.

2.5.1 Directed and Undirected

Directed graph or *digraph* is a graph where each edge has a direction. For $E = (V_s, V_t)$ an edge between source vertex V_s and terminal vertex V_t. In directed graphs, edges are

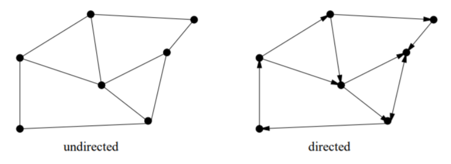


Figure 2.2: Directed graph versus Undirected graph.

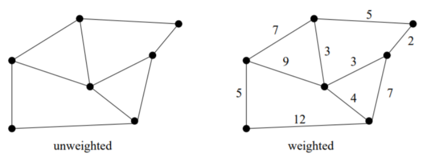


Figure 2.3: Weighted graph versus Unweighted graph.

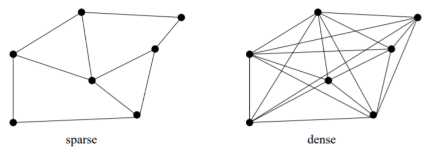


Figure 2.4: Sparse graph versus Dense graph.

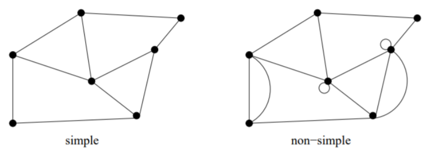


Figure 2.5: Simple graph versus Multigraph (non-simple).

represented by arrows. Each node here has in-degree $d_{in}(V)$ and outdegree $d_{out}(V)$ where the graph is called balanced graph when $d_{in}(V)$=$d_{out}(V)$ for all the nodes in the graph. While the Undirected graph is the graph that represents the relationship in a symmetric form, this imply that bond can exist in either (or both) directions.

2.5.2 Weighted and Unweighted

A graph in which edges are assigned with weights (i.e. real numbers) to indicate their strength. The importance of an edge for unweighted graphs are merely defined from their relationship with other edges.

2.5.3 Simple and Multigraph

A simple graph is a graph that has no self loops from the vertex to itself or multiple edges between the same vertices. While the non-simple graph (multigraph) allows self loops and multiple edges between the same vertices. Thus, the degree of a vertex in simple graph is the number of adjacent vertices to it while in multigraph the degree is the number of times it appears in the edge set.

2.5.4 Sparse and Dense

A graph is sparse when a small number of vertices have edges actually defined between them. The sparsity of the graph always depends on its application, for example, road networks have to be sparse due to the constraints by road junctions. Sparse and dense are sometimes used informally, but it can be made formal. For example, sparse graphs are linear in their size, however, dense graphs have a quadratic number of edges.

2.5.5 Connected

A connected graph is a graph where every vertex can be reached from any other vertex by traversing an infinite number of edges.

2.6 Graph Representation

If we are going to deal with the graph as a data structure, then, we need to define a way to represent it in memory. There are a couple of different methods that we can use, but we must keep in consideration that whatever the method we want to employ, our ultimate goal for all the methods is check the existence of an edge between two vertices. A common task in a graph is iterating over vertices adjacent to each other.

2.6.1 Adjacency Matrix

The adjacency matrix is a way to represent the edges of a graph. A graph can be completely determined by its adjacency matrix where the properties of this matrix are closely correspond to the properties of the graph. An adjacency matrix $A(G)$ is a matrix of size $n \times n$ where $A_{i,j} = 1$ iff vertex i is connected to vertex j and $A_{i,j} = 0$ if otherwise. This representation could be normally extended to represent digraphs, where $A_{i,j} = 1$ iff i and j were connected by an edge directed from i to j. The adjacency matrix for an undirected graph is always symmetric across its diagonal. It is usually preferred when the graph is dense and can represents either directed or undirected graphs. Many types of information about a graph can be easily derived with the help of adjacency matrix.

2.6.2 Edge List

Another common way to represent graphs is through a list of edges. Each edge consists of two vertices represented by just having one array of two vertices or linked list that store pairs of vertices. This can be useful if the edges are sparse in the graph.

2.6.3 Adjacency List

A graph represented with an adjacency list combines both edge list and adjaceny matrix. For each vertex in the graph there is an array of the other vertices adjacent to it, each of which contains a list of all adjacent vertices in an arbitrary order. The way we look for an edge (i,j) within the graph is that we check the i^{th} adjacency list when we go for the j^{th} in the i^{th} list. It is usually used to represent sparse graphs.

2.7 Special Graphs

- A **Complete graph** is a simple graph K_n having all possible edges between n vertices (every vertex is adjacent to every other vertex) where n = 2,3,4 ...... and $E = n(n-1)/2$.

- A **Bipartite graph** is a graph composed of two disjoint graph vertices sets such that no two vertices in the same set are adjacent. It is a special case of K-bipartite when K=2. A graph G is called bipartite, if V_G has a partition to two subsets X and Y such that each edge $ij \in G$ connects a vertex of X and a vertex of Y. In this case, (X, Y) is a bipartition of G, and G is (X, Y)-bipartite. There are variety of useful theorems about Bipartite graphs, for example:

  Theorem 2.7.1 *A graph G is bipartite iff $V(G)$ has a partition to two stable sets.*

  Theorem 2.7.2 *A graph G is bipartite iff every cycle is of even length.*

  Theorem 2.7.3 *A graph is bipartite iff it contains no odd cyles.*

- A **complete bipartite graph** $K_{m,n}$ is the graph that has its vertex set partitioned into two sets: X and Y of m, n vertices respectively. There exists an edge between two vertices iff one vertex from set X and the other vertex from set Y

- An **Acyclic graph** is a graph with no cycles. This type of graph contain at minimum one node with zero in-degree.

- A **Tree** is a connected undirected graph having no cycles. There are many results regarding trees. For example, a spanning tree is a subgraph that is a tree which includes all of the vertices of the graph G with the minimum possible number of edges.

  Theorem 2.7.4 *Each connected graph has a spanning tree that is a spanning graph that is a tree.*

- Other existing types like **star** and **ring** graphs [8].

2.8 Hamiltonian Cycles and Euler Circuits

In this section, we will present some definitions that Euler has used to show his theorem (discussed earlier in this chapter) to solve the problem of how to walk through the town and traverse all the bridges only once.

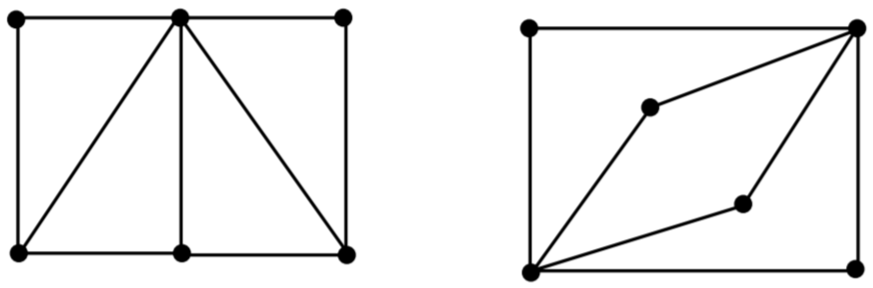


Figure 2.6: The left graph is Hamiltonian but non-Eulerian and the right graph is Eulerian but non-Hamiltonian.

- **Euler cycle** is a cycle containing every edge in the graph precisely once. A graph has an *Euler cycle* iff every vertex is of even degree. A graph G is said is Eulerian when it has *Euler cycle.*
- **Euler trail** is a path through a graph containing each edge precisely once, starting and finishing at different points. A graph has an *Euler trail* iff it has precisely two vertices of odd degree. A graph of this kind is called traversable graph.

  Theorem 2.8.1 *An Eulerian trail exists in a connected graph iff there are either no odd vertices or two odd vertices.*
- The **Hamiltonian cycle** is a cycle containing every vertex of the graph precisely once.

  Generally as shown in Figure 2.6, the core difference between an Eulerian cycle and Hamiltonian cycle is that, the first traverses every edge in a graph exactly once, but may repeat vertices, while the second visits each vertex in a graph exactly once but may repeat edges.

2.9 Graph Topological Cases

2.9.1 Isomorphism and Homeomorphism

1. **Isomorphism**

   An isomorphism exists between two graphs G and H if G and H clearly have the same structure and only differ in the names of vertices and edges.

   Theorem 2.9.1 *Two graphs G and H are isomorphic iff they have a common adjacency matrix and the two isomorphic graphs have exactly the same set of adjacency matrices.*

   If two graphs G and H are identical then they can have same diagrams. But, if G and H are non-identical graphs, they can also have identical diagrams. These graphs are not identical one but they are isomorphic graphs. They look exactly the same but their vertices and edges differ in their labels. This means that G and H are acquiring the same structure and the only difference in their names. The following definitions on isomorphism in graphs were presented in [9].

- Two graphs G=(V, E), H=(V', E') are said to be isomorphic if there is a bijection, such that this mapping $V(G)$ -> $V(H)$, $E(G)$->$E(H)$ is an isomorphism, i.e., iff i , j are adjacent in G the $f(i)$, $f(j)$ in G' are also adjacent.
- If two graphs are isomorphic then the subgraphs induced by the sets of vertices of a given degree are also isomorphic.
- If two graphs are isomorphic then they have the same number of cycles of a given length.

2. **Homeomorphism**

   A graph F is homeomorphic from a graph G if it can be obtained by subdividing edges of G or if they are isomorphic or they are both homeomorphic from a third graph H.

2.9.2 Planarity

The graph that can be drawn in the plane with no edge crossing so that its edges intersect only at their ends. Any planar graph can be colored with four colors. Long time ago a formula 2.9.2 discovered by Euler in 1736 to test whether a graph is planar or not. This formula plays an observable role in the study of planar graph. After that, Kuratowski discovered a criterion for a graph to be planar 2.9.2. Then, Whitney developed some properties of how to embed the graph into the plane.

- *Euler's Theorem*

  Let G be simple planar graph with V vertices, E edges and R regions then

$$V - E + R = 2 \tag{2.1}$$

- *Kuratowski's Theorem*

  Theorem 2.9.2 *The graph is planar iff it contains no subgraph homeomorphic from K_5 nor $K_{3,3}$.*

  The K_5 is the complete graph of order 5 while the $K_{3,3}$ is the complete bipartite 3 by 3 which is also named *Thomsen graph.* K_5 and $K_{3,3}$ are the fundamental blocks in a non planar graph.

where V is the number of vertices , E is the number of edges and R is the number of regions. Regions should be drawn in a planar form. If G is a simple planar graph then the degree of each region is at least 3. It should be also clear that a graph is planar iff each of its block is planar.

- *Maximal planar*

  The graph in which no more edges can be added without losing neither the simplicity nor the planarity. These graphs are some times called triangulated (every region is of degree 3).

  Theorem 2.9.3 *If G is a maximal planar graph then:*

  $$E = 3V - 6 \tag{2.2}$$

  Corollary 2.9.1 *If G is planar graph then:*

  $$E \leq 3V - 6 \tag{2.3}$$

2.10 Connectivity and Components

A **strongly connected graph** is a graph whose each node is connected to every other node through a direct path. The **components** of a graph G are the maximal connected subgraphs. A maximal connected subgraph is achieved when no any vertex or edge can be added and have a connected piece.

Sometimes the removal of a vertex or an edge can affect the number of components this is called a *cut-vertex* or *cut-edge* respectively. A cut-vertex is a single vertex whose removal disconnects the graph and the cut-edge is a single edge whose removal disconnect the graph. Sometimes the cut-edge is called a *bridge*. Anytime a graph contain a cut-edge then there is a cut-vertex but not vice verse.

A **block** is a maximal 2-connected subgraph. A block is 2-connected when it posses no cut-vertex or cut-edge. A **minimal block** is a block such that the removal of any edge e results in a subgraph $G - e$ such that $G - e$ is not a block. Every minimal block contain at least one vertex of degree 2. Hence G is not a minimal block.

2.11 Graph Coloring

In this section we define some of the fundamental notions for graph coloring. A coloring of a graph is one of the most essential concepts within graph theory. A proper graph coloring is the assignment of the minimum number of colors to the vertices or the edges of the graph such that no two adjacent vertices or two edges meet at the same vertex have the same color. We denote to this as vertex-coloring and edge-coloring respectively. This minimum color is known as **chromtaic number**.

2.11.1 Four Color Problem

Around 1850, the four color problem was conjectured, where a country map should be colored given four colors only. This problem involves the association of colors such that any two regions with common side should be given a different color. It has been known that four colors can color some maps and five colors can be sufficient for all maps [10].

2.11.2 Vertex Coloring

A K-coloring or k-vertex coloring of a graph $G = (V,E)$ is mapping f : V_G to $[1,k]$. A graph G is K-colorable given that there exists a proper K-coloring.

A **vertex chromatic number** $\chi(G)$ of G is the minimum number K for a graph for which there exists a K *coloring*. It is defined as:

$$\chi(G) = \min(k|exists\, proper\, k\, coloring) \tag{2.4}$$

therefore, if $\chi(G)$=1 then G is k-chromatic.

Theorem 2.11.1 *Let G be a graph, then $\chi(G) \leq \Delta(G) + 1$.*

where $\Delta(G)$ is the maximum degree of the vertices for a graph G.

2.11.3 Edge Coloring

An **edge coloring** of a graph G is the assignment of K colors to edges of G such that no two adjacent edges have the same color. An **edge chromatic number** $\chi'(G)$ of G is the minimum number K for a graph for which there exists a K edge *colorable*. There are some special cases for edge coloring according to the type of the graph.

2.12 Summary

In this chapter we introduce some graph theory concepts which we make use of some of them in the rest of the thesis and provide an overview of graph theory as presented in standard textbooks. The overview in this chapter is the mathematical introduction for the next chapter in which we will discuss in more detail the way of dealing with graphs, graph analytics and their application in social networks.

Chapter 3

Overview on Social Network Analytics

In this chapter, we introduce the foundational aspects of Social Networks Analysis (SNA), provide technical research work, applications in this area and discuss a set of research challenges related to mining data in social networks. The area of social network analysis is evolving fast in different directions so we will try, in this chapter, to cover the main aspects related to areas such as definition, approaches, mining and challenges. We will focus on analysis measures for individuals or the whole network in Section 3.5, as that will be needed for the rest of the thesis. In addition to giving an overview of popular mining areas in Section 3.6. Finally, we will end up highlighting different key problems and challenges of social network analysis which will be considered in the next chapters.

3.1 Social and Non-Social networks

There is a key difference between a social network and non-social network [11]. A social network focuses on the structure of relationships between social entities (Individuals) while a non-social network can represents countries, machines, products or any non-social entities. Analyzing a social network is based on an encapsulated concept which not only consider the individuals but also include the relational and actionable links between them. Wasserman [12] highlighted four main features of social network analysis as follows:

1. *Actors and their actions are viewed as interdependent rather than independent, autonomous units.*
2. *Relational ties (linkages) between actors are channels for flow of resources (either material or non-material).*
3. *Network models focusing on individuals view the network structural environment as providing opportunities for or constraints on individual action.*
4. *Network models conceptualize structure (social, economic, political, and so forth) as lasting patterns of relations among actors.*

Popular social platforms like Facebook, LinkedIn, etc. contain huge amount of data, information and lots of interactions for people within these networks and this has gained the attention of many researchers. Collecting such a huge amount of data, implied the shift to “computational social science” [13] with the availability of technological techniques that contribute significant advances in the research of social network.

3.2 Social Network Analysis

Early social network analysis was mainly based on graph theory, which has been proved to be very efficient on large networks. This basis of graph theory has proven to be a very powerful methodology to study and analyse physical or social networks. The roots of social network theory begin from social science, in addition to statistics and mathematics. The focus on actors, relations and patterns of existing relations requires a set of tools, methods and analytical concepts that are distinct from the methods of traditional statistics and data analysis. Thus, social network analysis is defined as the process of analysing social network and defining key actors, groups and relationships as well as changes in these variables [14]. While social Network analysis tools are a set of techniques, tools, methods and visualization techniques used in social network analysis which include graph theory concepts and statistical techniques [14]. Analyzing social network data is used in many fields to discover the social nature and behaviors of people in the network. In the recent years, social networks analysis has acquired huge attention due to the significant increase of social networks, accompanied by the availability of rich sources of social network data. This dramatic growth in social networks has encouraged massive number of users to share information, communicate, create new content and, based on their interactions, they can get useful recommendations. This rapid increase opened new challenges and unlocked the boundaries of studies and research for new analytics solutions.

There are two different kinds of approaches to analyze the network data [15]. The first approach, is the *socio-centric* approach which considers interaction of a group in the whole network where it comprises obtaining all relationships among a set of people in the network. The analysis in this case is based on techniques of complete networks such as groups analysis or measures that require a complete network structure. While the second approach, is the *ego-centric* approach which considers the network of only one individual. It asses the network quality of the person (density, size, etc.) and the attributes related to the ego network. The focus in analyzing social networks can be on investigating the relationships between individuals and the nature of their structure or can be on studying specific characteristics of these individuals [12].

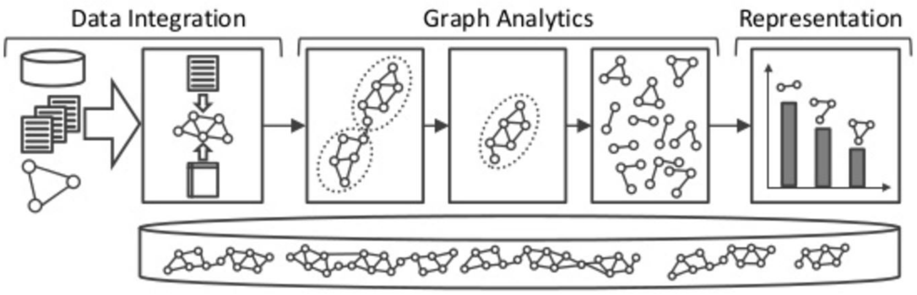


Figure 3.1: A full cycle of Social Network Analytics (SNA) process from importing data to results visualization.

A full network analytics cycle, from end to end, can be summarized in three stages that show the sequence of steps required for the analysis process (Figure. 3.1):

1. **Data Integration:** is the process of collection and integration of data from different data sources (e.g. graph databases, files, etc.) into a graph structure.
2. **Graph Analytics:** is the process of analytics which includes path analysis, connectivity analysis, community analysis and centrality analysis to discover the structure of the network (graph) or behavior of people within the network.
3. **Representation:** is the process visualizing the results of the analysis results and might be in from of charts, bars or visualized graphs.

3.3 Social Network Data Integration

Social networks provide for the sharing of huge amount of information between actors. Real examples of social networks can include: a group of employees in a large organization, with links joining people who work on the same project; a set of scientists in a particular field, with links people who co-authored papers; or a set of leaders in a business, with links represent people who worked together within a directors board. Some social network data gathered by researchers has been made available on-line to be used by other researchers. For example, Zachary's karate club dataset [16] is a popular freely available on-line dataset, where nodes correspond to members and edges correspond to interaction between those members. Another example is the Football network [17], where nodes correspond to teams and edges indicate if they were involved in a match against each other or not. Others real world networks such as the Les Miserables [18] or the Dolphins network [19] can be found in [20] or via Newman's home page [21]. As for the on-line social networks like Twitter or Facebook, a crawling process is frequently applied. This is the process to extract the user profiles from the network one by one using crawling techniques like Breadth First Search (BFS) and Depth First Search (DFS) [22]. Other data can be collected from different sources, such as relational databases, or graph databases and can be in different forms, either as structured or unstructured data files.

3.4 Social Network Graphs

Different mathematical methods are used to represent the network data for the analysis process. Both matrices and graph theory act as concrete foundations for many concepts in the analysis of social networks. Many researchers use the graph notation as a starting point for the analysis of a network. This plays a substantial role in understanding the data within the network and studying the results of the analysis. Graph representation includes three different types of data: (a) the nodes which can correspond to individuals, organizations or groups, (b) the edges which represent the relationship across the nodes and (c) the attributes which represent the different characteristics or properties either for the nodes or the edges. In addition to, these graphs capture the nature (e.g. density) and the direction of relationships (e.g. directed, undirected).

Models based on graphs have been used frequently to do analytics on social networks using various types of graph representation. Different types of graphs such as directed, undirected and weighted depend on the kind of the network itself. For a Facebook network, if a user becomes friend with another, then both agreed to have a relationship, therefore the network is considered undirected. While for a Twitter network, a user can follow another one without having his consent and without being followed back from the other user. This case defines the direction of the edge. The weights in some type of networks play an essential role in assessing the quality of the relation as they reflect the frequency and the level of interaction, for example, the frequency of email contacts between two actors in an email network.

Representing social networks as graphs is known as socio-grams in the field of sociology. Combining these graphs with some concepts and measures from graph theory can provide a visualized description for the data and the information within the social network, especially for small size graphs. For large graphs, the high density of edges make it difficult to provide an informative visual picture.

Generally, graph theory is essential for analyzing social networks. It is used in the analysis of these networks to determine the most common properties and features of the network, of the nodes and of the links. It clarifies some important insights about actors; influencers, experts and trustworthy people [23] or active, engaged and initiator actors in the network [24], this in return will have a great influence on activities and benefit opinions or even decisions made by decision makers or management.

3.5 Network Analytics using Graphs

In this section we clarify how the graph theory concepts, which were presented in the previous chapter, are used as analytical measures for a social network.

We give the definition of the most popular metrics that can be used for analyzing any network and which are used in our experiments in the next chapters. We cover the local properties that corresponds to the individual and some global properties that corresponds to the network as a whole.

3.5.1 Preliminaries

Analysis of social networks can be applied in any field that can be modeled as a graph $G = (V, E)$. These graphs represent a rich source of information. In on-line social networks, the vertices (nodes) represent actors and the links (edges) represent interactions or relationships among actors. The edges between the nodes can either be directed, which means that there is a source node and target node for each link, or undirected, which means there is a edge between two nodes without specifying the source and the target. The degree of the node is the number of edges to other nodes in the graph. For directed networks, there may be a differentiation between in-degree (the number of coming edges) and out-degree (the number of leaving edges).

3.5.2 Topological Properties

The network properties are the properties that provide information about the structure of the network as a whole not to specific individuals in the network. They aggregate the entire relational and behavioral interactions between the individuals in the network. These measures can describe the size and the overall structure of the network. Other network properties relate to node and link properties and illustrate different connection patterns between people, identify the roles and highlight important people, groups and events. All of the following metrics can help in reporting and giving insights on leadership, health, organization and hierarchy. Here are definitions of some of popular properties which are used in many research works and in our thesis.

- **Size.** The size of the network is commonly used to find the way the network can be analyzed and crawled or obtained. It is defined as the number of nodes within the network. For instance, small sized network can easily be gathered and analyzed but on the other side, large sized networks need automated methods. Thus, size is considered an important feature that provide researchers with insight about the network. The type of the network is considered the main factor that affects the number of nodes in the network [22].

- **Diameter.** It is the longest path among all shortest paths calculated between actors. Diameter affects the speed of the diffusion of information within the network.
- **Small world.** One of the common properties in many networks. This property is used to refer to two properties: the *small distance*; when two people are joined through a short chain of reciprocal acquaintances and the *clustering effect*; when two people are likely to know each other when they share a common neighbor.
- **Degree Distribution.** An important feature of networks is how the links are distributed over the nodes in the network.
- **Power law.** The power law confirms that the number of vertices with degree k is proportional to $K^{-\gamma}$

  for some exponent $\gamma > 1$. The power involves using one parameter only (the exponent) to illustrate the distribution of degree for billions of nodes and facilitate applying a comprehensive analysis of these networks It is is a

  first-order estimate and an important basic to understand networks. The power law distribution can be illustrated in the normal scale or logarithmic scale. The precise distribution follow a power-law form [25]. Networks with power-law distribution are called scale-free networks because their degree distribution depends on properties other than the size of the network. The formal definition of the degree distribution $P(k)$ is the probability for a fraction of nodes in a network having degree k. A common property is popular in social networks which is the power-law degree distribution the is denoted as:

  $$P(K) \sim K^{-\gamma} \tag{3.1}$$

  Where k represents the degree , γ the exponent and $P(k)$ is the degree distribution.
- **Triads Count**: The aim of this algorithm is to count all the triangles or in other words the cliques of size 3. Counting the triads can be beneficial for many graph algorithms because they can be used to view similarity between structure of graphs [26] and can also be useful for detecting the communities (groups) in the network [27]. Hence counting triads is considered as a graph metric that is the basis of other analysis algorithms.
- **Centralization.** Centralization has a crucial importance in social networks analysis as it uncovers the persons who have critical positions within the network. For

instance, leaders and popular actors occupy central positions in the network. The most commonly used centrality measure in social network analysis community are degree, betweenness, closeness which were proposed by Freeman [28] and eigenvector centrality which was proposed by Bonacich [29].

1. *Degree Centrality* is the number of direct ties that involve a given node. It is based on the idea that an important node is the one with largest number of links relative to other nodes in the graph. According to Freeman, it is defined as:

   $$C_d(i) = \sum_{i=1}^{N} A_{ij} \tag{3.2}$$

   Degree centrality indicates the activity of the direct relations to the node i. N is the number of nodes in the network, A is the adjacency matrix where $A_{ij} = 1$ if there is a link between the nodes i and j and $A_{ij} = 0$ if there is not a link between these nodes. The measure focuses on the most observable actors in the network. Actors with low degree act as peripherals while actors with high degree act as a main channel of information within the network.

2. *Closeness Centrality* is a measure which is based on the minimum geodesic distance $d(i,j)$. The geodesic distance is the minimum length of an indirect path from i to j where $i \neq j$ according to Freeman definition distance.

   According to Freeman, it is defined as:

   $$C_c(i) = \sum_{j=1}^{N} \frac{1}{d(i,j)} \tag{3.3}$$

   Closeness centrality indicates the freedom of the node i to control actions of others. N is the number of nodes in the network and $d(i,j)$ is the distance between node i and other nodes. The measure focuses on how close an actor is to all other actors within the network. The idea of closeness is inversely proportion to distance. As the distance of a node from other nodes increases (more geodesics linking a node to other nodes), the closeness centrality will decreases.

3. *Betweenness Centrality* is the number of shortest paths between all the pairs of nodes that passes through this edge in the graph. It is based on the concept of minimum geodesic distance. According to Freeman, it is defined as:

   $$C_b(i) = \sum_{j \neq k} \frac{g_{jk}(i)}{g_{jk}} \tag{3.4}$$

   Where $g_{jk}(i)$ is the number of shortest paths between j and k passing through i and g_{jk} is the total number of shortest paths between j and k where $j \neq k$.

Betweenness centrality indicates the intermediate location of a node and its ability to limit or facilitate interactions within the nodes it links. The measure focuses that the actor must be between many of the other actors through geodesic distance.

4. *Eigenvector Centrality* is a positive eigen vector of adjacent matrix. According to Bonacich, the eigenvector centrality $C_e(i)$ of a node i is defined as:

$$C_e(i) = \frac{1}{\lambda} \sum_{j=1}^{N} A_{ij} C_e(j) \quad \forall i \tag{3.5}$$

   Where λ is a constant, N is the total number of nodes, A_{ij} is the adjacency-matrix of an undirected (connected) graph and $C_e(i)$, $C_e(j)$ are the scores of the i^{th} and the j^{th} node respectively.

A node is more central if it is in relation with nodes that are central themselves. Thus, it can be claimed that the centrality of a node not only depends on the number of its adjacent nodes, but also, on their value of centrality.

- **Detecting Communities.** A community or a cluster is defined as a group of nodes that have a better connection within a group and sparsely connected with other groups in the network. There are numerous community detection algorithms for finding communities within the network [30]. These algorithms assign nodes to communities when there is no available ground truth. An overview and a comparison will be presented in chapter 5 for some of the commonly used community detection algorithms.

- **Modularity.** This metric is proposed by Newman [31]. The *modularity* metric is used when examining communities with the network to evaluate and asses the quality of dividing the network into communities. Modularity ranges from -1 to 1, and is 0 when no community structure. Practically, a real network with significant community structure can have a modularity from 0.3 or more [31]. This metrric is defined in [32] as:

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{K_i K_j}{2m} \right) \delta(c_i, c_j) \tag{3.6}$$

  where A_{ij} is the adjacency matrix of the network between node i and j, K_i is the degree of node i and c_i is the community of node i and $\delta(u, v) = 1$ if $u = v$ and $\delta(u, v) = 0$ if otherwise.

- **Transitivity.** It is the average clustering coefficient of the all the nodes within the graph. It ranges between 0 to 1 where high values correspond to high cliquishness, clustering coefficient with value 1 corresponds to perfect cliques and clustering coefficient with 0 corresponds to no triangles found between connected nodes. It is defined as:

$$C = \frac{1}{V} \sum_{i \in V} C(i) \tag{3.7}$$

While the internal transitivity within a group or community depends on how the direct neighbors of a certain node are connected. It is the actual number of links (edges) between neighbors, divided by all the possible links if they are all connected.

The internal transitivity T of a community C is defined as:

$$T = \frac{1}{N_C} \sum_{i \epsilon C} \frac{2 * l(i)}{k_{in}(i)[k_{in}(i) - 1]} \tag{3.8}$$

where N_C is the number of nodes in community C, $l(i)$ is the number of actual links between neighbors of the node i and $K_{in}(i)$ is the indegree of the node i.

- **Edge Density** The edge density $\mathcal{P}$ of a community C of an unweighted graph is the ratio between the actual realized links in the community E_C to the maximum number of possible links it can contain if all the nodes are well connected to each other $N_C(N_C - 1)/2$ where N_C is the number of nodes in the community. Communities are supposed to be higher in density than the whole network.

  The density function is defined as:

$$\mathcal{P} = \frac{2E_C}{N_C(N_C - 1)} \tag{3.9}$$

- **Connected Components.** In the case of undirected graphs, a connected components is a subset of nodes so that there is a path between each pair of nodes. While for directed graph, a differentiation is made between a *strongly connected component* (SCC) and a *weakly connected component* (WCC). Strongly connected component corresponds to set of nodes where a path exists between all the pairs in the set. In contrast, weakly connected component corresponds to a set of nodes where a path exists between all pairs in the set if all the links were viewed as undirected in the network. Studies have shown that there is usually a dominant strongly connected component within the network [33].

3.6 Mining Social Network

As discussed before, social networks provide a wealth of information that can be transformed into valuable insights about the dynamic and the structure of the network. There are some typical questions researchers should ask when analyzing a social network. Some of these questions are: Who knows who in the network and how strong is the relationship? How well do people know each other's knowledge, expertise and skills? Who is the source of information and spread the word? What resources do people use to share information? How could the pattern of individual choices derives more holistic patterns, may be using predictive modeling (e.g. correlation and regression) on network data? Answering these types of questions is what has been known as *mining a social network*.

From our preservative, and based on the existing literature, we categorize the mining areas of social network into four popular categories: Node Mining, Link Mining, Content Mining and Community Mining, and provide a literature review of some of the existing work in each category.

3.6.1 Node Mining

Node mining is a type of user behavioral analysis to find patterns in the network, predict popularity, actions of actors [34], and find influencers [35]. It studies how actors are embedded and located in the overall network. Several studies have been focused on identifying the role of actors in the network [36]. A quantitative study of the topological characteristics is proposed in [37] to report the results of the participation of users within a Twitter network. The study includes ranking users to find the top influential people within the network based on their number of followers and page rank. Rowe [38] presented an algorithm based on mining an email network of the Enron corporation, one of the well-known publicly available email dataset for real corporation. His analysis focused on analyzing the behavior of users and finding the communications patterns to extract hierarchal levels that reflect organization work chart and define the main players within the network. [39] studied the period of Enron's crises to find the structural features of the network, extract structural properties and find the key players within this period. Varshney [40] analysed the level of response of employees based on email exchanges. Diesner [41] conducted a study to find the communication patterns via their identified hierarchical structure. Marlow [42] identifies authoritative blog authors by investigating to what extent the blogs are inter-connected.

3.6.2 Link Mining

Several studies have been focused on the quality of ties between actors [36]. There are numerous methods that are based on the number and the density of ties to find influential, reputable, authoritative and central people in the social network. The output of these methods is usually a ranking score which corresponds to a reputation or social prestige from the social network perspective. PageRank is one of the popular methods which is best known as it is used by Google to rank web pages [43]. It derives scores not only for specific nodes, but also for those connected to them. Gynogyi [44] proposed a method called TrustRank. The method derives a trust score for each node. Another method, introduced by [45] is called HITS. This method requires computing two scores for each node, the first is called *hub score* for the node with many outlinks and the second is *authority score* for the node with many inlinks. Generally, methods like PageRank, TrustRank and HITS have been used by many researchers to find "interesting" people in a social network [46]. They are also used to find expert people in the network [47]. Other ranking methods like betweenness centrality [48] and eigenvector centrality [49] are also commonly used.

Links or relationships in social networks often contain patterns that can correspond to properties like the rank or the importance of the object. Sometimes, it is desirable to predict the existence of a link when it is not observed or to predict if the link will come into existence over time as the network is evolving. This is why the link prediction problem has also gained attention in the literature. Link prediction is the problem of inferring missing links based on the observed network. The problem is formalized in [50]. They propose link prediction approach that concludes that future interaction can often be predicted based on the topology of the network and measures for detecting the proximity of nodes within the network.

3.6.3 Content Mining

Mining the content within the network is a way to discover useful information, content features and classify content into topics. For example, in [37], tweets are ranked and analysed based on their active duration to find the most trendy topics, where most of the topics appeared to be headline of news. Generally, this work covered several questions like: What topics do they talk about? How is the information diffusing within the network?. Leskovec et al. [51] shows that the dynamics of popular topics in online social networks are made up from succesive focus and defocus on topics and that's result into information diffusion in the networks. Mccallum et al. [52] discovers topics within discussion based on sender-recepient relationship in email network and this combines the connectivety within the network with topic clustering. Gloor et al. [53] aims to improve data quality and discovers insights within an enterprise through mining content in communication archives such as blogs, instant messages and emails.

3.6.4 Community Mining

Over the recent years Community detection has attracted attention of researchers enormously in terms of the different proposed community detection algorithms. A community can be group of people who may be interested in same topic and contribute to each other through post or replies. Identifying communities has a crucial value in network analysis. They can reveal a priori unknown information related to topics or cyber-communities. Tsagkias [54] focuses on prediction of community activity using eight different new networks and finds the pattern across each network. Rowe et al. [24] proposed an analysis of community types and based on dynamics of community behavior using one single network. Gibson [55] proposed a technique to derive hyper-linked communities in Web environments, which includes hubs and authorities identification which are discussed above in Subsection 3.6.2.

Generally, the description of network structure for complex networks has been studied in different ways in the field network analytics. Finding the community structure is considered in between two different levels. It can be *node-based* level which involve finding the properties of nodes (centrality, degree and so on) and can be *network-based* level which involve finding the whole network properties (clustering coefficient, degree distribution). Generally, the basic definition of community structure is introduced in [56] by Girvan and Newman where they defined it as “The division of network nodes into groups within which the network connections are dense, but between which are sparser”. Based on this definition, nodes should be densely connected to each other within the community while having few links to nodes in different communities. A hierarchal phenomena can even exist such that different levels of sub-communities can exist within each community.

There is no specific requirements for community size but it often follows the power-law distribution [57]. Leskovec [58] observed that in real networks, communities blend more with the rest of the network as the community size increase. This consequently reduces the appearance of communities and their quality. Thus, the focus is good to be mainly on the densely small sized communities. A very useful overview about community structure in networks is proposed by Fortunato in [59]. Besides, the dynamics and the evolution of communities which makes the analysis process of finding communities over time more challenging as it requires applying clustering algorithms at different timesteps which will results in independent communities that can be hard to link over time.

3.7 Network Analysis Challenges

Despite the potential benefits of analysing social networks, there are many challenges to mapping and analysing actual real world relationships. There are a wide range of challenges in the literature in this area that can be rich for research.

In this section, we tried to categorize the social network challenges from the literature by trying to highlight some common challenges in area of social networks and categorize them from our own perspective. These challenges will mostly relate to our contribution in the rest of the thesis. We will provide in the next chapters an in-detail discussions about these challenges and propose our contribution which we aim to unlock some challenges and can be a way forward in areas related to community structure, evaluation, and networks dynamics in social network analysis.

3.7.1 Data Collection and Preparation

Collecting and extraction of data can be done either manually through self-reporting and interviews or using sensors, web-crawling, automated network extraction methods, for example, relation extraction or entity extraction [60]. Collecting personal data is not free, and we want to make sure that we get the best value from the data. There are many challenges to collect activity within the network. Standardizing on methodology, models, and tooling could significantly reduce the effort, risk, and cost of collecting such data. It is a challenging process as it requires having a clear picture about the available data, its format, who have the rights over this data and if there is any missing or poorly documented relevant data. Even for available data there might be complexity in the structure of the data and its records may include many fields that are not relevant from social network analysis perspective. Additional problems might arise like: duplicate nodes, for instance, a single node with two different emails or a person who is no longer in the network or removed his account and his profile remain active. Therefore, data cleaning and pre-processing is a required step before the analysis.

3.7.2 Evaluation and Validation

Choosing an evaluation metric in a principled way is difficult, as often collecting and sharing data is not easy even if the data is available as reaching a ground truth for validation is difficult. This problem, in the data mining literature, is known as lack of ground truth. Besides, the novel techniques in social media research, there is a need for evaluation without a standard reference or ground truth.

Many researchers approach ground truth problem through surveys to identify knowledge flows and relationships to people to derive a hypothetical scenarios [23]. However these scenarios might indicates the way people think but not necessarily reflects their actual behavior. These kinds of techniques do not assure validation, but reduce the percentage of error during the analysis process and help in validating the insights to some extent.

3.7.3 Community Detection

One of the common problems and challenges in the field are community detection problems and their evaluation. Many researchers devoted their research work to discover communities in social networks (graph clustering). The problem arises in two ways: the first is ensuring the accuracy and the second is ensuring the quality of communities [61]. When ground truth is available, assuring accuracy is achieved when the objective is to check and compare to the actual communities. This can generate an accuracy score which can correspond to accuracy metrics [62]. Assuring both the accuracy and the quality is not an easy task if ground truth is not available [61]. As ensuring quality is achieved through measuring the feasibility of the community structure depends on the connectivity inside the community relative to connectivity to other objects outside the community within the network. Thus, the scores generated by quality metrics are based on the structure of the networks and the community structure within the network [63].
Several community detection approaches/algorithms have been proposed to find communities and groups in the network based on graph theory concepts. However, one of the main challenges is to find the optimal number of communities and the appropriate community structure for structureless networks or unstructured communities. There is challenge of comparing different community results and decide which one is the optimum with hidden community structure. There are various functions proposed with the aim to compare results or find the optimum one within a given network.

3.7.4 Evolving and Dynamic Networks

Many researchers have focused their attention on the evolution and the dynamics of social networks. It was observed that most of the networks turned to be much denser across time due to the time stamped datasets [64]. Consequently there is a super linear increase in the number of nodes and the number of edges within these networks [65]. A lot of work focused on this dynamic change in the data which can reveal a new type of information and uncover the interaction between communities. Another issue that is getting more popular is the study of graphs evolving in time. This is now possible due to the availability of timestamped network data sets. Tracking the evolution of community structure in time is very important, to uncover how communities are generated and how they interact with each other. Several steps have been taken in consideration with the evolution of data which will lead consequently to the evolution of graphs to achieve an efficient mining process. However, there is a common problem which is the lack or the high cost of big data technologies and having insufficient computing resources or clusters that can handle the huge size of datasets. Despite the fact that there are many open source and web formats that can help in the preparation of this data but there is always a problem facing the management of large scale datasets.

3.7.5 Blinding Decision

Active networks are the networks that represent active engagement of actors in the network (tags, likes, comment, etc). This active participation can shape and influence the network structure or the relation between nodes. The more active ties or engagement in the network, the more information and insights derived. The active relations play a substantial role in breaking down traditional hierarchies and silos as well as informing insights or decisions. While passive relations such as friendship, follow, etc. within the network may result into information loss between its nodes and might lead to a potential problem of deriving insights about people. For instance, an employee in an organization might have a network of friends, but he might be not a fan of using social networks. If decisions will be based on the network structure only, then this will blind having real insights about this person. Trying to friend many people as possible does not reflect the real communications and ties. Social media in some cases does not reflect the actual social interactions and experience. Passive behaviors (browsing, reading other's stuff, etc.) or passive relations (friend, follow, etc.) may not be a real interaction. These types of relations just acknowledge that you share a space with others. Another problem which is the articulation of the right boundaries of the network through filtering and selection, defining the boundaries and the partiality of the network (who is in and who is out) actually matters to reflect a qualitatively informed understanding of the nature and the characteristics of the social network structure. Also, missing relations or missing nodes in the network can affect the derived decisions as they hide information about their effect on the whole structure of the network.

3.7.6 Privacy and Ethics

Despite the fact that collecting data from on-line and off-line sources is much easier than before, but, still there are many challenges facing researchers when they need to use data for development or for management the streaming of data. Preparation of data is an essential step as discussed before not only for efficiency, but also, for anonymization process to overcome privacy issues. Some common new challenges like computational complexity, security issues arise when revealing a sensitive and confidential information about people or organization. Data mining paradigms have been proposed to perform mining tasks taking into account data protection to preserve privacy of data or personal information. Networks or graphs can be rich sources of data that discovers and reveals information about personal identity and insights about users. Defining the right balance between hiding data and disclosing data is suggested through many approaches. This approaches include auditing queries [66] and sanitization of data [67]. Removing names or identification number from the data is not sufficient, as the structure of these graphs can reveal and reflect information about the individuals themselves [68]. The main challenge is how to anonimize data by hiding personal information or sensitive structure in addition to having a useful data to recover useful insights. Data protection can impact our ability to collect personal data from the network, that's why mechanisms are needed to model, collect, and manage consent of data.

In addition to privacy, there are ethical and legal issues. For example, ethical policies in an enterprise, laws related to data across countries or even from industry to another industry. All of these cases wil have severe impacts and limitations on the analysis process and what exactly can be done on the data.

3.8 Summary

After presenting a background on the graph theory concepts in the previous chapter which are used as a mathematical base for the work proposed in this thesis. In this Chapter, we focused on presenting an overview of the social network analysis. First, we introduced the definitions and the concepts of social network analysis. Then, we discussed the type of data presented in social networks and how this data can be represented in social graphs. Also, we reviewed the analytics that can be used for mining social networks and discussed the different areas of mining. Finally, a special focus has been given to the challenges in the social network analysis area which will be the core of our contribution in the next chapters.

Chapter 4

Analysing and Predicting the runtime of Social Graphs

In this chapter, we introduce our predictive technique that can address the evolving graphs problem in the area of social networks. The model provides an estimated execution time for the end user indicating the analysis that can be done on a given network without requiring any information about the network except the number of nodes and edges.

4.1 Motivation

The explosion of Social Network Analysis in many different areas and the growing need for powerful data analysis has emphasized the importance of in-memory big data processing in computer systems. Particularly, large-scale graphs are gaining much more attention due to their wide range of application. This rise, accompanied by a massive number of vertices and edges, led computations to become increasingly expensive and time consuming. That is why there is a move towards distributed systems or Big Data cluster(s) to provide the required computational power and memory to handle such demand of huge graphs. Thus, figuring out whether a new social graph dataset can be processed successfully on a personal machine or there is a need for a distributed system or big-memory machine is still an interesting question. In this chapter, we try to address this question by providing a comparative analysis for the performance of two of the most well known SNA tools: Gremlin [85] and SNAP [86] for performing commonly used graph algorithms such as counting Triads, calculating Degree Distribution and finding Clusters which can give an indication of the possibility of carrying out the work on a personal machine. Based on these measurements, we train different supervised machine learning models for predicting the execution time of these algorithms. We compare the accuracy of the different machine learning models and provide the details of the most accurate model that can be exploited by end users to better estimate the execution time expected for processing new social graphs on a personal machine.

4.2 Evolving Graphs Problem

In recent years, there has been an observed increase in the number of large on-line social networks, many of them have a massive number of users that can reach hundreds of millions of users [69]. Analysing these social network databases can provide rich information which can be beneficial for a wide range of applications and areas but is sometimes considered expensive and time consuming. This is due to the expected super-linear increase in the computational time and therefore speed and scalability should be key challenges of social network analysis.

In order to handle huge real-world network analysis problems, distributed clusters may be required to accommodate "real-world" graph sizes. Alternatively, big-memory machines that can do a highly interactive analysis that can have advantages over distributed clusters [70]. A long computational time may be needed to handle any graph analytics like community detection, node ranking, computing shortest paths, number of triads, degree distribution and connected components. The need to have an efficient computational tool/model or even a query language to use with this social graphs has been addressed by many researchers e.g. [69], [71], [72].

What if we have an option to run our analytics on different computational platforms? How could we predict which platform is most suitable? This will be the goal of our work in this chapter, to be able to predict the execution time of graph algorithms for unseen graphs using two of the most commonly social network analysis tools as an example, but to reach this predictive execution time we will need first to know how currently available tools perform on a personal machine. There are many social network tools and libraries that can perform a set of operations, features and various algorithms with many functionalities for graph analysis of this rich data and information within the graphs. For example, SNAP[1], Gephi[2], NetwrokX[3] and Gremlin[4] are some of the most commonly used tools in the community.

4.3 Big Data and Graph Analytics

Perez et al. in [70] proposed Ringo, a big-memory graph analytics tool, which supports interactive graph analytics of millions of edges through merging big-memory machines that can outperform all other distributed systems. The authors showed that a single machine with big-memory can provide an efficient platform for doing graph analytics. Distributed graph systems like Pregel that support parallel graph algorithms on multiple

[1] http://snap.stanford.edu/snap
[2] https://gephi.org
[3] https://networkx.github.io
[4] https://github.com/tinkerpop/gremlin/wiki

machines and support adoption of a Bulk Synchronous Parallel (BSP) model was proposed by Malewicz et al. in [73] and also GraphLab in [74], a distributed system for data mining and machine learning. In [75] Kyrola et al. presented the performance of GraphChi, a disk-based system on a PC that supports evolving graphs overtime, GraphChi has a low memory requirement which was designed specially for computation on big-scale graphs. The authors performed a comparison between GraphChi and other distributed systems like Spark [76], Hadoop [77], PowerGraph [78] and GraphLab, it was found that PowerGraph can compute graph analytics using large cluster much more faster than using GraphChi on just a single machine. The comparison was performed on PageRank, one of the most popular graph algorithms. It was shown that GraphChi can provide high performance for different purposes.

Seo et al. in [79] claimed that the performance of Datalog, a declarative logic programming language that is usually used as a query language [80], is not competitive with other low-level languages in the past. However, it allows the expression of many graph algorithms and supports recursion and high-level semantics which consequently allow optimization in time and parallelization. A high level query language for graph analytics named SociaLite was proposed by the authors as a Datalog extension for powerful analysis on graphs. The authors performed a comparison for the execution times for running a shortest path graph algorithm on different benchmarks like Giraph [81] and Hadoop and then compared their execution time with SociaLite concluding that the latter outperforms. The authors presented a comparison for the execution time in [69] between Datalog engines like Overlog [82], IRIS [83] and LogicBlox [84] for running shortest path algorithm on single machine, showing a better performance for LogicBox. However when compared with SociaLite, the latter showed a better performance than LogicBlox. Also, the authors proposed a comparison of SociaLite with other implementations in java of almost 50%.

We have found that tackling the performance issue to predict an estimated time needed for analysing graphs is a new area that can be fruitful for detecting the execution time of evolving graphs. To the best of our knowledge, this is the first comparative analysis that aims to find an estimate prediction for the execution time of graph analytics based on different benchmarks using a PC.

4.4 Experimental Components

In this section, we introduce the tools, the measures and the predictive models used in the experiments. We give a brief description of each of them and we highlight the purpose of selecting each of them in our technique.

4.4.1 Investigated Tools

As we descibed, the increase of Social Network Analysis is driven by the rise of on-line networks specially human networks [12]. This rise has driven many researchers and developers to create and develop different approaches, algorithms and tools to easily apply graph mining and analytics. This led to having a plentiful supply of publicly available frameworks that have many algorithms supporting the study and manipulation of data for any type of network. Our main concern will be how to select the right tool for the observed large-scale evolving graphs and decide which tool can suit your system design, graph size or even the algorithms that are to be used.

Our experiments will target a comparison between examples of two dfferent types of graph analytics tools. We have chosen to conduct this comparison between Query-Language-Based tools like Gremlin and Software-Based tools like SNAP. Query language tools are based on query language which can be used for generic purposes and enable many users to do social network queries in an easy and professional way without having a software background [69]. While, other Software tools like what mentioned before can be C++ and Python-based, so they are supposed to have a better computational time. We concisely summarize the features of both tools as below:

- **Query Based Tools**

  Gremlin is an example of a Query-Based tool. It is a domain specific language for working with graphs, a graph based programming language developed for multi-relational graphs, named property graphs. The following are the main features of Gremlin:
  1) Supports complex graph traversals.
  2) Works over different frameworks, graph databases and graph processors.
  3) Used within the Java language as a virtual machine that has a direct access to Java based application.
  4) Combines query language, network analysis and manipulation of graphs.
  5) Enables a wide range of users who do not have a software background to do efficient and easy queries.

- **Software Based Tools**

  SNAP is an example of a typical Software-Based tool. It is a free general purpose network analysis and graph mining based package tool with the following features:
  1) It is written in C++.
  2) Provides a Python interface (snap.py) [87] for use with Python and runs on Win-

dows, Mac OS and Linux.

3) Scales to huge networks with hundreds of million of nodes and edges.

4) Calculates the graph's structural properties, provide standard graph algorithms and different network structure measures.

4.4.2 Graph Measures

We tested the execution time of three popular graph analysis algorithms which were discussed in detail in chapter 3 for both tools.

- **Triads Count**: The aim of this measure is to count all the triangles or in other words the cliques of size 3. Counting the triads can be beneficial for many graph algorithms because they can be used to view similarity between structure of graphs [26] and can also be useful in community detection [27].
- **Degree Distribution**: This is a simple measure to count the number of edges for each node in the graph, it is based on the concept of neighborhood to find the vertices that have the most direct links to other vertices.
- **Detecting Clusters**: The aim of this measure is to find communities (clusters) or know how many unique clusters and what is the distribution of vertices within each cluster.

The analysis of these graph measures and their execution time are implemented using Python language for SNAP tool and Groovy[5] scripts with Gremlin[6] language for Gremlin tool.

4.4.3 Predictive Modeling

With the continuous growth of data in various social graphs, learning how to take decisions based on this data to improve business or provide solutions is an important need. Consequently, learning a model for the performance issues using the number of nodes and number of edges as predictors for the model can be useful for making decisions on where to run a graph analysis job. That is our main reason for choosing to apply supervised machine learning algorithms to learn how to quickly a system can perform graph analytics.

Supervised learning is a useful and popular type of machine learning. There are several types of supervised machine learning algorithms that have been presented within the Machine Learning area like classification, regression, and anomaly detection [88]. Supervised machine learning algorithms are used to make predictions based on a set of features. They

[5] http://groovy-lang.org/

[6] http://gremlindocs.spmallette.documentup.com/

are used to find patterns in the data. Each algorithm looks for different types of patterns. After the algorithm has derived the best pattern, then this pattern is used to provide predictions for unlabeled testing data. Supervised learning can be applied on any type of data (e.g. financial, seasonal, geopolitical) and it has several applications [89].

According to [88], it was concluded that the following are the most suitable regression models that can be used to address this type of problem: Support Vector Machine (SVM), Multivariate Adaptive Regression Splines (MARS), M5 and Boosting. These are used in our experiments to train the results from the performance analysis in order to select the best model that will predict the execution time of unseen graph based on two main features that are usually known in any social graph: nodes and edges. Our target is to present an approximate model as a computational technique that provides a relation between the execution time and the structure of the network.

- **SVM**: The SVM model [88] is considered for both regression and classification problems based on detecting whether the data can be categorized or not. It is closely related to robust regression through reducing the effects of outliers in the regression process. Also, SVM is based on the value of the linear combination of the input features and capable for representing non-linear relationships in a linear fashion using a kernel function (RBF), which is a method for using a linear algorithm to solve non-linear problems.
- **MARS**: The MARS model [88] usually uncovers important data patterns effectively, it is more flexible than other regression models and does not require any data preparation. The nature of MARS features is that it splits the predictor into two groups and then start to model the linear relationships between the outcome in each group and the predictor. The MARS model can be interpreted easily through the existence of the *hinge function* which partitions the input data automatically and works more appropriate for numeric variables make them work efficiently for numeric data. A pair of *hinge function* is usually written as $h(x - a)$ and $h(a - x)$.
- **M5**: The M5 [88] is a model tree algorithm. It is used for the approximation and modeling complex non-linear problems. M5 is regarded as a promising model for prediction of numerical problems and is popular because of its robustness and efficiency where it can tackle tasks with very high dimensionality. The main implementation of this model is included in the Weka software package [90].
- **Boosting**: The Boosting model [88] is known to be a powerful predicting model. It combines the outputs of multiple weak regression models to obtain a better model for a final prediction. The algorithm is considered as a powerful prediction tool as it

usually outperforms other individual models. The algorithm of the boosting model gets initialized at first with a best guess (e.g., the mean value) and then the gradient is calculated, then, a model is fit to minimize what is called as *loss function* and the current model is added to the previous one. This process is repeated according to the number of iterations specified by the user.

R scripts are implemented to apply predective modeling using R language [91] and Caret [92] library to train and test the performance results of the graph measures for SNAP and Gremlin. For tuning the parametres of the four models, we used the *trainControl* function which computes the default parameters for each model. As for the method used in each model, *svmRadial* method is used for SVM with RBF kernel function, *earth method* is used for MARS, *M5* is used for model tree and *gbm* is used for Boosting. Then, the selection of the models is assesed by measuring the **Root Mean Square Error (RMSE)**. It is a commonly used error metric to measure the performance of regression models. A linear regression model is fit with least squares, which means minimizing the RMSE.

4.5 Experimental Setup

In this section, we start by outlining our testing setup and environment. Then we give an overview of the dataset used and our terminology in preparing and preprocessing the data for the experiments. We provide a comparative analysis for the results based on the SNAP and Gremlin tools and how their computational times differ for the same graph metrics.

4.5.1 Setup

- **Test Machine**: Most of the experiments were done on an Apple Mac Pro computer, 2.4 GHz Intel core 2 Duo (2 cores), 3 MB cache size and 8 GB of main memory.
- **Test Protocol**: Each test is performed under the same setup and configuration. In the tests, we used multiple iterations and the mean execution time was reported. The datasets were held in-memory for each test. All our experiments were based on undirected graphs. It is worth noticing that the computational timing measured for both tools does not include the time taken to load the file into memory or writing the results into a file.

4.5.2 Facebook Dataset

The dataset considered in this chapter is a public available Facebook dataset collected by Stanford University [20], a freely available real world graph dataset. This dataset represents a list of friends from Facebook, it presents a political affiliation social graph between members. The data was anonymized be replacing the internal ids for each user by new value. The circles consist of 4039 vertices and 88234 edges. Each vertex represents a user and an edge exists if any two users have same political affiliations. To perform our experiments, we divided this data into subgraphs to compare the results on different sizes of graphs with growing number of nodes and edges. We will explain the subgraphing methodology in Section 4.6.

4.6 Evolving Graph Preparation

In order to test the performance of the tools and to simulate the evolution phenomena using one graph, a subgraphing process took place in order to extract subgraphs of varying sizes from the complete graph. We divided the Facebook dataset into subgraphs, each subgraph was represented as a selected number of nodes and all their associated edges or links between them in the whole network. The selection of the nodes IDs is based on their ID value in the graph. For instance, with a subgraph of 100 nodes, we select the nodes with ID values between 0 and 100. We extracted 10 subgraphs using snap.py library [87], by the main graph and specified nodes IDs in vector form as their parameters and returns a subgraph induced by the nodes specified in this vector and the edges between these nodes. We repeated this process for having different subgraph sizes. The resulting subgraphs are 10 times smaller in edge count and nearly 4 times smaller in node count of the whole graph. Our strategy selected: $100, 200, 300, ...1000$ nodes and their associate edges so the network size varies from 100 to 1000 nodes and their edges numbers varies from 275 to 9890 edges as shown in Figure 4.1. Hence, we evaluated the computational time on a graph growing constantly. Given this is a Facebook graph, the graph type is undirected with no multiple edges or self loops. These subgraphs were represented as an edge list in different files ready for analysis.

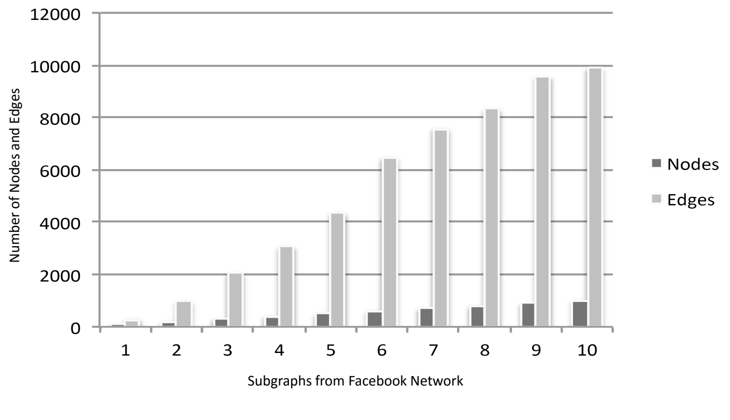


Figure 4.1: Subgraphing process for Facebook Dataset.

4.7 Results and Discussion

In this section, we provide a discussion of the results for the performance analysis that has been conducted using different graph measures using two tools as discussed above. Then we presented the process of training and testing this performance analysis using four different supervised machine learning models as described above. We then asses the output of the model and evaluate them using RMSE measure.

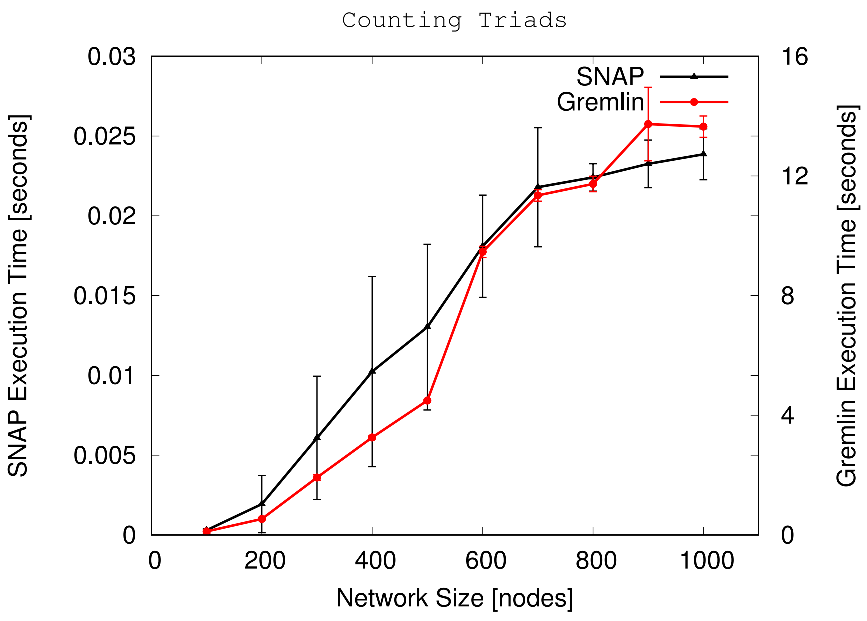


Figure 4.2: Execution time of counting triads for SNAP and Gremlin represented on different scales.

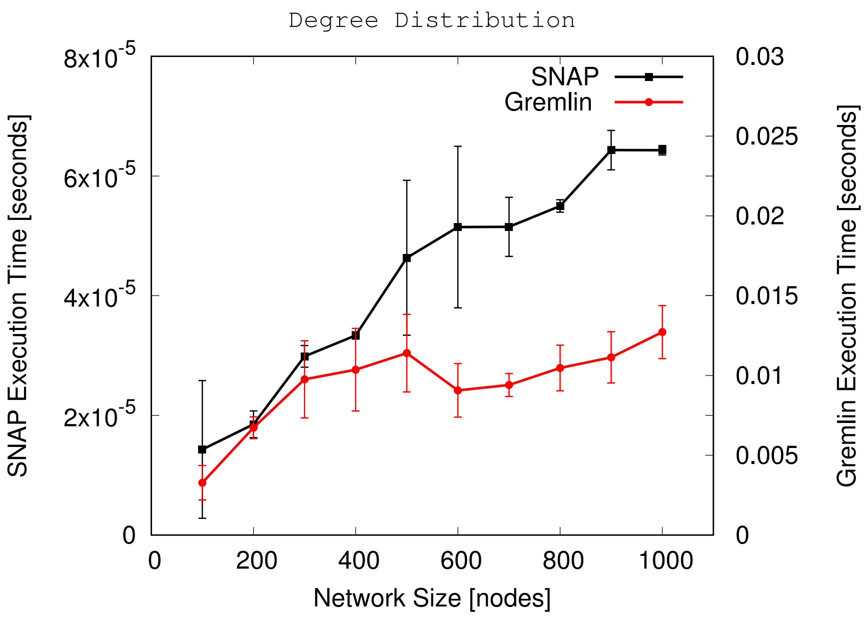


Figure 4.3: Execution time of degree distribution for SNAP and Gremlin represented on different scales.

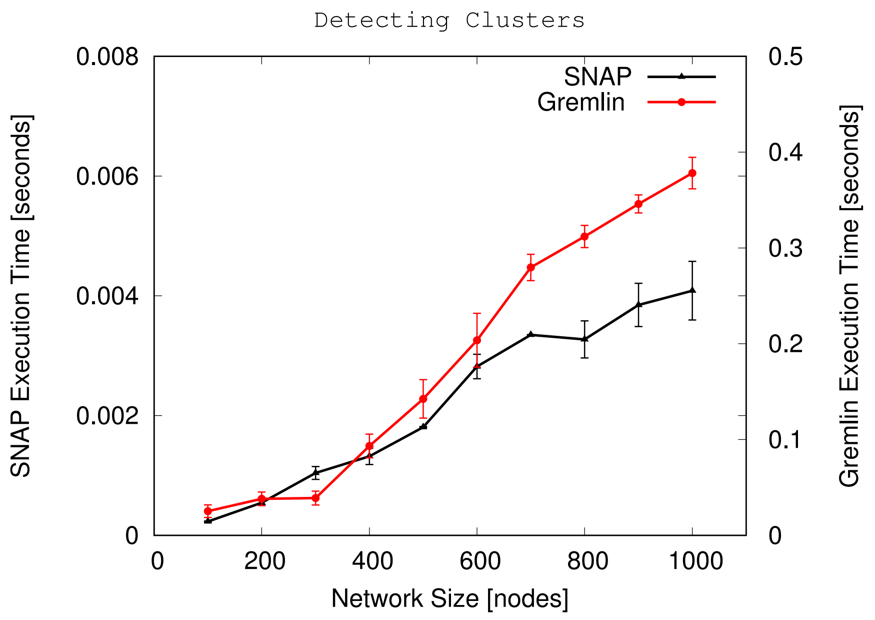


Figure 4.4: Execution time of detecting clusters for SNAP and Gremlin represented on different scales.

4.7.1 Performance Results

This subsection presents the performance results of the tools discussed in the previous section. The comparison is based on the execution time measured by both tools to calculate different metrics on different graph sizes. We expect that the computational time will be affected by the structure and the size of the graph. The elapsed time for running the algorithms will differ based on whether these algorithms access the edge list one or more times.

Referring to Figures 4.2, 4.3 and 4.4, we measured the execution time taken by each tool to measure each of Triad Count, Degree Distribution and Detecting Clusters on different sizes of subgraphs. For the sake of increasing the accuracy of our reported results, we repeated each measurement for the execution time 20 times. The reason for the variance indicated by the error bars is seems to be due to runtime performance variability of the software and hardware. We repeated these tests with the same subgraph size by selecting different nodes to build the subgraph and found the execution time was similar. This indicates that the mean execution time is not affected much by the nodes selected and that the variation arises from runtime issues.

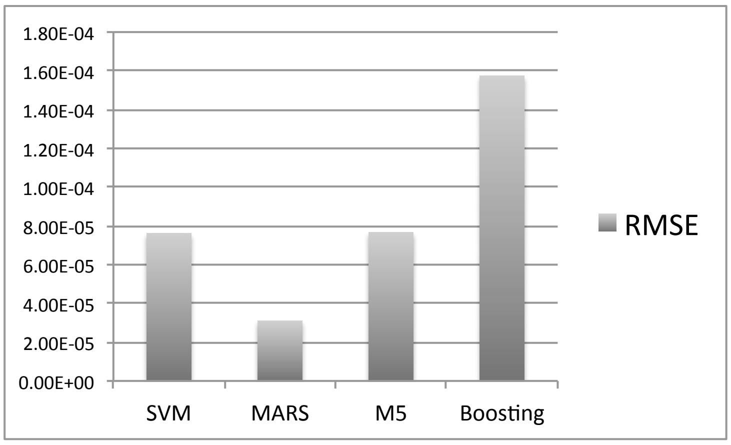
In the first test, for the *Triads Count* as shown in Figure 4.2, we found a major difference in the performance of both tools. SNAP performed much better than Gremlin, the execution time taken by Gremlin was nearly 13 seconds to traverse 1000 nodes and 9800 edges while in SNAP it took nearly around 0.025 seconds and this is because the query used in Gremlin for counting the triads is likely touches and traverses every vertex in the graph to check their connection with their neighborhood and then check that their neighbors are connected so this led to traversing many vertices more than once. Counting triads of large-scale graphs usually require a fast algorithm, specially for graphs having billions of nodes and edges and it is preferable to be in a parallelized processes.

For the second test, the *Degree Distribution* as shown in Figure 4.3, both tools performed better compared to calculating the triads. Unexpectedly, we observed that initially the execution time of Gremlin was high for the small subgraphs then it started to decrease when approaching a graph size of 500 nodes. It is worth emphasizing that we repeated the same experiment multiple times but while the dip is within the error bars, there does seem to be a trend. We do not have a clear justification of this behavior. The Gremlin query here traversed all the vertices to get their edge count. As for SNAP, it performed normally with an observed linear increase with the size of the network.

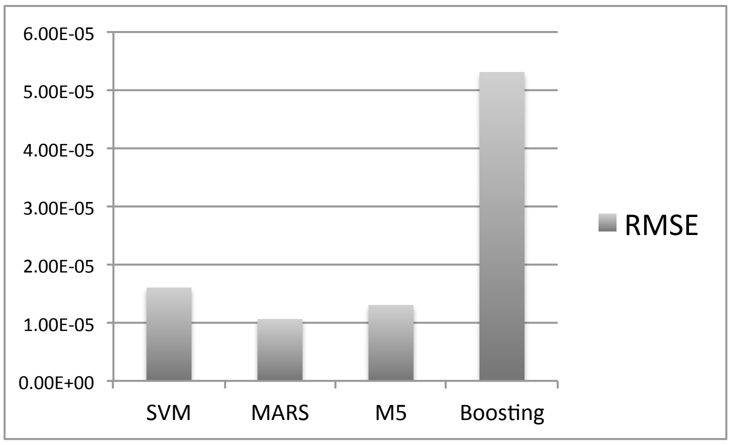
In the last test for *Detecting Clusters* as shown in Figure 4.4, we observed that using SNAP took around 0.04 seconds for a graph with 1000 nodes while Gremlin took 0.4 seconds for the same graph so it is clear that SNAP is 10 times faster than Gremlin. The algorithm used in SNAP is based on computing the average clustering coefficient for small-world networks as defined by Watts and Strogatz [93]. While, the Gremlin query used in this test is based on the peer pressure vertex program algorithm, where every vertex assigned what is called by nominal value and if two vertices have same value therefore they are in same cluster and acquire the same cluster ID. Overall, SNAP performs much better than Gremlin in the three experiments.

4.7.2 Training Performance Results for Prediction

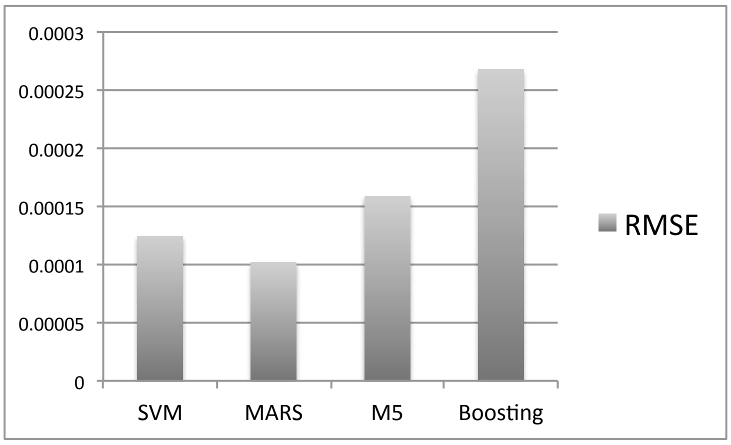
For the sake of training the machine learning models and achieving better accuracy, we extended our measurements to 50 various sizes of the same dataset and calculated the execution time for each of the three algorithms for both tools. In order to train and test the models, a general practice is to split the data into a *training set* (75 % of the dataset) and *test set* (25 % of the dataset). We applied one split method on our dataset formed from the 50 samples to a training set of 35 samples and 15 samples for the test set. We used the training set for estimating the coefficients of the different machine learning models whilst the test set was used as a test data for evaluating their performance. The resulting performance profile appeared to have an observable difference between the four models in terms of Root Mean Square Error (RMSE).



(a) Counting Triads



(b) Degree Distribution

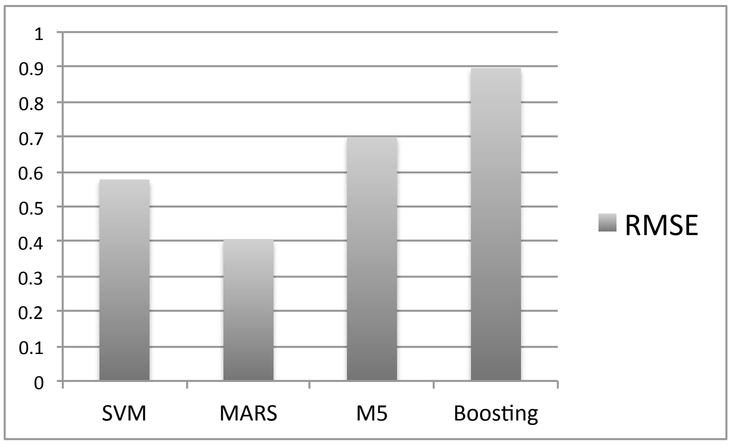


(c) Detecting Clusters

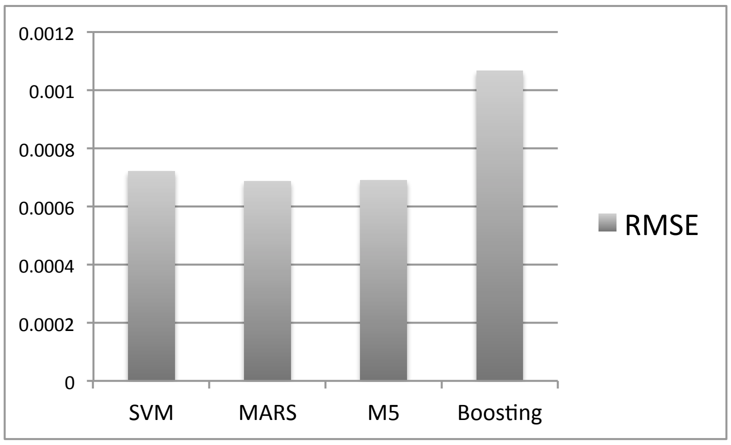
Figure 4.5: RMSE for predicting the execution time (in seconds) for the three metrics using SNAP.

4.7.3 Prediction Results

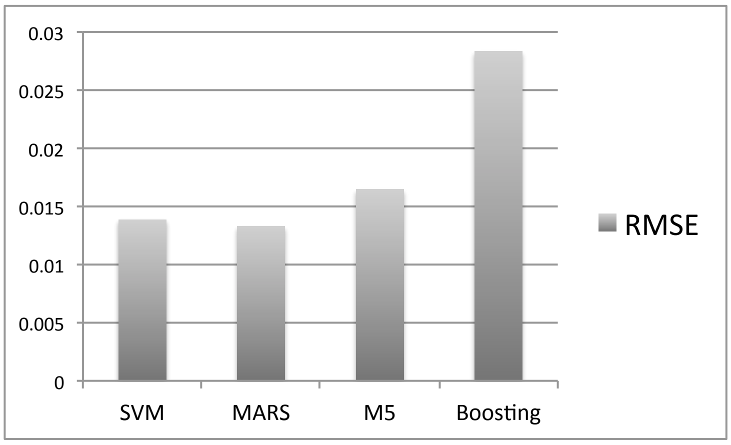
Referring to Figure 4.5 and Figure 4.6 , the graphs represent the RMSE for each model for predicting the execution time for each of the Counting Triads, Degree Distribution and Detecting Clusters using SNAP and Gremlin as shown in the figures. It is clear that for both tools, the Boosting model had the highest RMSE for all of the three metrics. On the other hand, we found that the best model with the minimum RMSE for both tools



(a) Counting Triads



(b) Degree Distribution



(c) Detecting Clusters

Figure 4.6: RMSE for predicting the execution time (in seconds) for the three metrics using Gremlin.

regarding the three graph metrics is MARS. For SVM and M5, the first outperformed the latter for *Triads Count* and *Detecting Clusters* while on the other side M5 outperformed for calculating the *Degree Distribution* for both SNAP and Gremlin. Therefore, our proposed prediction will be based on MARS model since it showed the best performance for all metrics using both tools. Hence, we derived our MARS based model for predicting the execution time based on the number of nodes and edges for measuring each metric illustrated by Equation 4.1 where the coefficients (a, b, c, d, and e) along with the hinge functions (h_1, h_2, h_3, and h_4) for each metric for both tools are defined in Table 4.4, Table 4.2 for SNAP and Table 4.3, Table 4.1 for Gremlin. We denoted by **ST**, **SD** and **SC** as the Counting Triads, Degree Distribution and Clusters respectively for SNAP. Similarly, **GT**, **GD** and **GC** for same metrics but for Gremlin.

$$Time = a + b * h_1(x) + c * h_2(x) + d * h_3(x) + e * h_4(x)$$
$$where\ h_{1,2,3,4}(x) = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases} \tag{4.1}$$

	a	b	c	d	e
GT	-1.05	0.009	-0.012	-0.002	0.002
GD	2.7	-6.7	1.4	0	0
GC	0.05	-0.00009	0.0005	-0.0009	0.00008

Table 4.1: Coefficients of the MARS model for the execution time using Gremlin.

	a	b	c	d	e
ST	-1.5*10^{-4}	1.7*10^{-6}	-1.5*10^{-6}	-4.3*10^{-7}	3.8
SD	2.9*10^{-4}	-2.9*10^{-7}	2.1*10^{-7}	-7.5*10^{-8}	2.7*10^{-8}
SC	0.003	-0.0000004	0.0000002	0	0

Table 4.2: Coefficients of the MARS model for the execution time using SNAP.

	$h_1(x)$	$h_2(x)$	$h_3(x)$	$h_4(x)$
GT	h(500-N)	h(N-740)	h(1522-E)	h(E-1522)
GD	h(340-N)	h(N-340)	0	0
GC	h(340-N)	h(N-340)	h(N-820)	h(E-8620)

Table 4.3: Hinge function coefficients of the MARS model for the execution time for Gremlin.

	$h_1(x)$	$h_2(x)$	$h_3(x)$	$h_4(x)$
ST	h(500-N)	h(N-500)	h(1522-E)	h(E-1522)
SD	h(420-N)	h(N-420)	h(1522-E)	h(E-1522)
SC	h(7153-E)	h(E-7153)	0	0

Table 4.4: Hinge function coefficients of the MARS model for the execution time for SNAP.

4.8 Summary

This chapter first propose a performance comparative analysis between social network analysis tools using a personal machine. Our results related to two different types of tools: Software and Query based tools, SNAP and Gremlin respectively. Gremlin tool showed lower performance than SNAP, especially in calculating the number of triads in the graph. This suggests that using Gremlin should be accompanied by having a cluster to be able to parallelize many computations. On the other hand, SNAP performed efficiently on a personal machine and showed better results for all the metrics, so it can be useful for anyone who is comfortable with Python or C language. While SNAP can provide analytics on massive graphs it may lack features related to compatibility issues with graph databases and graph processors which are supported by Gremlin. Next, in order to provide the end user with a prediction tool for the execution time, we trained different machine learning models: SVM, MARS, M5 and Boosting on our test data to help take good decisions that facilitate timely analysis of the graphs. They report the execution time of different graph sizes using three graph metrics for SNAP and Gremlin. Our experiments concluded that MARS outperformed other models for both tools. Hence, we provided a computational formula based on MARS model for each graph metric for both tools to estimate the execution time needed to analyse a given graph.

Chapter 5

The Impact of Algorithm and Interaction on Communities

In this chapter we fully exploit the methodology behind different community detection algorithms and study the delivered partitions of different types of communications within the Enron email network. The chapter includes a comprehensive comparative study using topological measures of the derived communities within the partitions. In addition to, discussing different evaluation measures to asses the quality of the partitions.

5.1 Motivation

There are various approaches to the task of comparing sets of communities. One of the obvious ones is to compare the results with the ground truth. Another approach is to compare the results of different community algorithms to each other. It is worth emphasizing that we can not always compare communities to ground truth, for example, the Enron data lacks the ground truth for communities. However, our main contribution in this chapter is to study and analyse how social community structure and the quality of partitioning change over various network relations and algorithms. We will do this by quantifying the resulting communities' structural properties and by applying different quality measurements to elucidate the issue of the community structure differences using multiple views.

5.2 Introduction

In the real-world, similarities or connections between entities can be determined by various relationships/interactions. Relationship can be friendship between actors (e.g., family, business, school) or how they communicate (e.g., email, mobile, text). These kind of relations in social networks are represented in what is called a "social graphs" as described in chapter 3, where edges here represent different types of relationships between actors. For example, in the Enron email network, different relationships/interactions can be: who cc'd, sends to or bcc'd who in the network. Each of these represents the type of the exchanged emails in the network. We will see that deriving different graphs from the same network, where each one corresponds to a specific type of interaction, can affect the derived communities. We will denote to these interaction-based graphs in our study as **Partial Network Views** where each view models a different and specific interaction within the network.

As we mentioned in chapter 3, community detection in social graphs is one of the considerable interests and challenges that have also acquired great attention in the research (e.g. [30]). In fact, it has been targeted by many studies and considered as an important part in the area of *Network Analysis*. The reason for this is that communities reveal necessary understanding for analyzing the behavior of people with each other within the network. In networks/graphs, a community can be defined as a group of nodes that have a better connection within a group and sparsely connected with other groups in the network or in other words, nodes that have better internal connections than the external ones [30]. Therefore, algorithms exist to detect communities which are denser in connection, smaller in structure and have strong connections within its vertices. Generally, most communities which have dense links are most likely to have common properties, that is why the concept of similarity is linked to the concept of community. Hence, similarity measures have a great influence on detecting communities [94].

There are various approaches for community detection algorithms that have been presented and studied in the research community. Each algorithm has its own approach in defining communities. Some of the different approaches are based on: random walks, spectral analysis, label propagation, centrality, modularity and many others [95]. These algorithms can be compared from different perspectives, either from the process that leads to finding the community structure or from studying the community structure itself. Choosing the best algorithm that can suit specific problem is not an easy task [96]. In this work we focus on studying and exploring the following questions:

1. *How similar are the results of different community detection algorithms?*

2. *How does the partial network views concept affect the derived communities?*

3. *Which type of network view leads to more informative communities?*

In summary, we further study the change in communities under the change of two factors: first, the partial network views and second, the community detection algorithms.

5.3 Community Detection Approaches

We will now review the community detection schemes to identify those that will be used in our analysis for this chapter and the next chapter.

As mentioned in chapter 3, the informal definition of the community as a group with densely connected edges has been common. However, there are numerous definitions or approaches for algorithms that implement different strategies for finding the community structure. Approaches can be based on optimizing modularity, like FastGreedy [31], Louvain [97] and Spinglass [98], or algorithms like Leading Eigenvector [99] and Commfind [100], which are spectral algorithms. Others algorithms that are based on random walks like MarcovCluster [101] and Walktrap [102], or information theoretic algorithms, like Infomod [103] or Infomap [104]. In this section, we will review the different approaches of community detection and their algorithms however, we will discuss the methodology and the family for each of the used algorithms in this chapter and in Chapter 6.

5.3.1 Node Similarity Based Algorithms

This category is based on the similarity measures between nodes, such that a community is group of nodes which are similar to each other and dissimilar to the rest of the network. A cluster analysis, based on similarity [105], is applied once all the node-to-node similarity are detected. In our work, we used the Walktrap algorithm from this category.

Walktrap (WT) is a random walk based algorithm where hierarchical agglomerative clustering is the applied approach proposed by Pons and Latapy [102]. Random walks mean that at each step the algorithm moves from one node to another through a random choice. Generally, the idea is to use the distance measure from one node to another to identify communities. For example, if two nodes j and k are in same community, then the probability to a third random node i to be in the same community should not be that different from both j and k. This algorithm uses a node similarity approach where each community is detected as a group of nodes similar to each other and dissimilar from the rest of the nodes in the network.

5.3.2 Compression Based Algorithms

This category is based on data compression to derive the communities. They consider the whole network represented in a compact way. The derived communities depend on maximizing the compactness and minimizing the loss of information. We used the Infomap algorithm from this category.

Infomap (IM) is an example of compression-based approach. It is based on information theoretic principles. The community structure here is derived based on Huffman coding [106]. It tries to minimize or compress the information quantity over the network.

This approach does not use the separation and cohesion concepts like other community detection definitions.

5.3.3 Link Centrality Based Algorithms

The type of algorithms in this category rely on link-centrality measures which are based on a hierarchical approach. At first, the algorithm deals with the whole network as one single community. Then, all the central links between communities are repeatedly removed until the network splits into several components. The process is repeated until a finer community structure is reached. We selected one algorithm from this category.

Edge Betweenness (EB) is proposed by Girvan and Newman [17]. It is a hierarchical process where the edges are removed from the network in a decreasing order according to their edge betweenness scores. It measures the centrality of a specific edge by finding the percentage of shortest paths passing through this edge in the network and this highlights the importance of the edge. The algorithm yields a good results but is not commonly used for large-scale graphs due to its high computational complexity. This type of algorithm is based on measures of centrality approaches. They deal with the network as one entity where the network splits into multiple components by repeatedly removing central edges.

5.3.4 Neighborhood Based Algorithms

This category is based on the concept of node neighborhood and diffusion of information within the network to derive communities. We used in our work Label propagation algorithm from this family.

Label Propagation (LP) uses the neighborhood concept and assigns each node to one unique label from k labels. Then an iterative process takes place where each node is assigned the label that is mostly common in its neighborhood. The process stops when each node has the label that is the most frequent in its neighborhood. Communities are then constructed by targeting groups of nodes having the same label [107].

5.3.5 Modularity Based Algorithms

As defined in chapter 3, modularity is the most wide spread optimization function for deriving communities in the network. The measure asses the separation and the cohesion through deriving the number of inter and intra community links [108]. This category is based on deriving the communities that are clearly separated from the rest of the network and cohesively connected from inside. Therefore, they combine two opposite aspects; separation and cohesion. One algorithm from this category (Louvain) will be used in our experiments for chapter 6.

Louvain (LV) is based on the modularity optimization. It adopts the agglomerative hierarchical method with an additional aggregation step for dealing with large networks [97] which differentiate it from the greedy optimization approach [31]. The algorithm consist of two steps, the first step includes placing each node in its own community to maximize the modularity then there is a check for each node, the node moves into a community which have the highest modularity gain from its neighbors' community or remains in the current community if there is no possible gain, the process is repeated until no further improvement can take place. The second step is based on building a new network whose nodes are the derived communities from the first step. Then, the first step is repeated again on this network until stable communities are achieved.

5.4 Topological Properties of Communities

In this section, we study some popular topological properties of communities that were defined in the literature [109], [110]. Measures like size distribution, density, transitivity (refer to chapter 3) can deliver description about the interactions and the topology of communities. They are used to compare and characterize the community structures but not to evaluate community detection algorithms.

5.4.1 Size

The size of the community is the overall number of communities including communities with single vertex derived by a community detection algorithm.

5.4.2 Size distribution

The distribution of the community size is one of the important features of the community structure. They are often unevenly distributed and sometimes obey a power law [111] with exponent ranging between 1 to 2 [27]. Mostly, the minimum size of a community in real networks is usually 2 while the maximum size can vary in a wide range depending on the used model [112].

5.4.3 Singleton

A singleton community is a community that contains only a single vertex.

5.4.4 Transitivity

The transitivity depends on how the direct neighbors of a certain node are connected. It is the actual number of links (edges) between neighbors, divided by all the possible links if they are all connected.

5.4.5 Edge Density

The edge density of a community is the ratio between the actual realized links in the community to the maximum number of possible links it can contain if all the nodes are well connected to each other. Communities are supposed to be higher in density than the whole network.

5.5 Performance of community detection algorithms

The traditional techniques used for assessing the performance of community detection algorithms deal with the community structure as a partition of the node set. Evaluating and comparing partitions is a classical problem in the area of community detection. The comparing process involves comparing the estimated community structure with a reference one. They compare the communities taking into consideration only the individual node membership, unlike the previous topological measures which take links into account. There are many ways of comparing the similarity between partitions. But, since there is no one single quality measure for comparing communities from different algorithms [113], therefore, in this section we will review some of the most commonly used measures that have been presented in the literature. Some of these measures are based on global metric like *Modularity* which compares the results relative to random graphs or based on counting pairs like *Random Index* or *Adjusted Random Index* and others are based on the use of mutual information like *Normalized Mutual Information.* The general strategy for comparing partitions as shown in Figure 5.1 where $P_1 = C_1^{'}, C_2^{'}, ..C_n^{'}$ and $P_2 = C_1^{''}, C_2^{''}, ..C_m^{''}$ are examples of two sets of communities from different partitions.

5.5.1 Modularity

Modularity [108] is one of the most commonly used method to evaluate the quality of partitioning a network into communities. It is used to measure the quality of division within a network into communities. It is a common measure used to determine and compare the performance of community detection algorithms. chapter 3 showed how modularity is calculated.

5.5.2 Random Index (RI)

The Random Index [62] measures the agreement for a given pair of nodes to be in same community for the estimated and reference partition. It counts the pairs of nodes that are classified correctly. The *RI* ranges from 0 (when pair misclassified) and 1 (correctly classified) under different partitions. The *RI* depends on the number of nodes that are

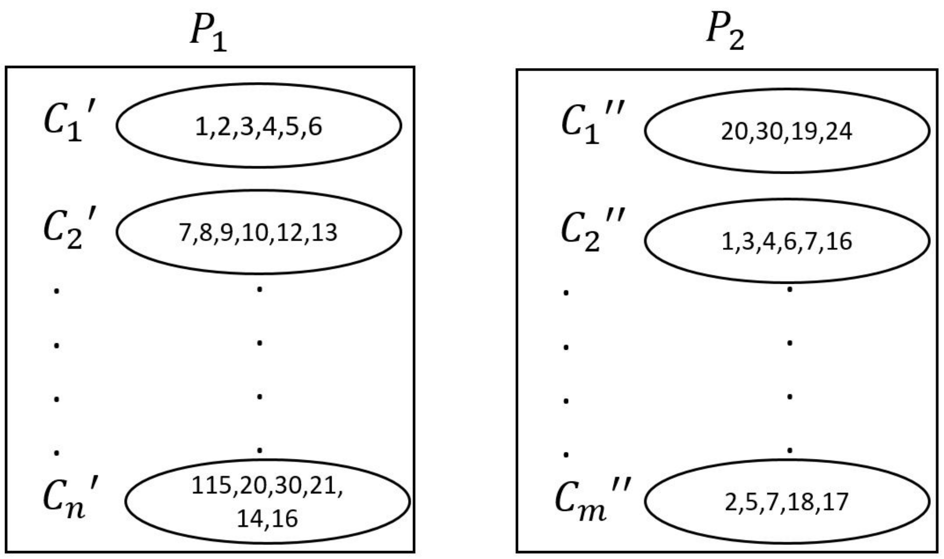


Figure 5.1: Non-overlapping communities derived from different community detection algorithms on the same data.

grouped in the same community for partitions P_1 and P_2 and the number of nodes that are grouped in different community for both partitions.

5.5.3 Adjusted Random Index (ARI)

This is an adjusted version of RI [114] that is proposed by Hubert and Arabie. The ARI is the normalized difference of RI and the value expected under a null hypothesis (the number of clusters be the same in the two clustering). It compares how much two partitions have common information between each other using the same dataset. The measure takes the value 1 when the resulting partition perfectly matches the reference one and takes value 0 when the algorithm fails completely to detect a matching community structure.

5.5.4 Normalized Mutual Information (NMI)

NMI [115] is one of the classical measures that ranges from 0 to 1 when perfectly corresponds to the reference partition. It was proposed for the classical clustering to compare different partitions for one dataset. It compares how much common information between two different partitions. It is used in the research community [109] to measure the performance of community detection algorithms. The measure was used by many authors [116], [109] to asses the quality of community detection algorithms.

5.6 Enron Email Corprus

Enron[1] is a dataset that has large number of emails for individuals of Enron's staff. It is a fertile data source for a real corporation that was collected by Melinda Gervasio at SRI within the period 1998–2002. The data was made publicly available by the *Federal Energy Regulatory Commission.* The Enron corpus is considered the largest real email data in the public domain. William Cohen made the dataset available publicly on-line for researchers. The data is a rich source that can be ideal for text analysis, social network analysis and link analysis as well. The Enron data has different types of emails either personal or official, thus, some of the emails have been deleted according to the request of employees. The dataset that is on-line contains around 151 employees and their email logs recorded from December 1999 to June 2001. The logs include information like Message Id, From, To, Subject and email content. There are no attachments included in the log files. The Enron data is organized in 3500 folders. Each employee has a personal folder and inside each folder sub-folders for the emails sent, deleted and junk. The staff are mostly from the management level.

5.6.1 Enron SQL Database

For this email corpus a network, a database schema is formulated by Shetty and Adibi [117]. The database consists of four tables, each one represents a different entity: employees, messages, recipients and references. They cleaned the emails by deleting any unneeded ones, duplicates or even blank ones and fixing aliasing problems.

5.6.2 Enron Data Preparation

We used the SQL database schema proposed by Shetty and Adibi to build our own Enron SQL database. Then we began to filter and clean the data according to the needs our experiments. The database represents two types of information: the first is the communication type between the employees and what we mean by communication type is the *TO*, *CC* or *BCC* fields, while, the second type is the content of the messages in the emails. Our work will focus on the first type of information. We focus on extracting the communications between the employees. We extracted the "From" and "To" , "From" and "CC" fields of the emails in order to be able to build sender and receiver email lists. We deleted all personal emails, any emails outside the organization and identified users who have more than one email address.

[1]https://www.cs.cmu.edu/ enron/

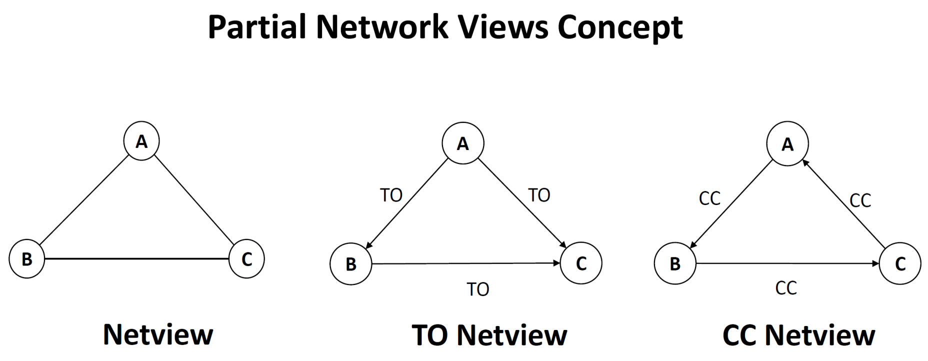


Figure 5.2: Partial Network Views from Enron Network.

5.7 Different Network Views Concept

Our focus is to derive three different communications network views as we illustrated before. Nodes will represent Enron employees and edges between them correspond to a specific type of communication, which will differentiate between the views. We model each type as a graph where each corresponds to a partial view in addition to a general view that only indicates that there is an existing edge whatever the type of communication.

As shown in Figure 5.2, the first view will represent an *undirected graph*, we refer to this network as **Netview**, a single edge will exist here between any two emails if there is any type of exchange in emails between them. The second view is based on the *TO* field in the emails, we represent this as a *directed graph* that reflects the direction of communication of the emails, we denote to this graph as **TO Netview**. For the third view, edges correspond to the *CC* field in emails, we represent a *directed graph* and refer to it as **CC Netview**. We note that we were interested in studying the *BCC* communications but these were not available in the dataset.

As shown in Table 5.1, we explored some generic properties for each view like: *nodes*, *edges*, *cliques* (nodes that are tightly connected to each other) and *clustering coefficient* (the degree of nodes that tend to cluster with each other). We found that an employee named “Paul Barbo” is missing from the three network views, also, for the *CC Netview* around six employees do not appear to have any communication (Mary Fischer, Steven Merris, Joe Stephenovitch, Joe Quenet, Andrew Lewis, Paul Barbo). The three networks views have almost the same clustering coefficient (≈ 0.3). The number of cliques is bigger in the undirected network (*Netview*) and remains the same for both directed networks (*TO Netview*, *CC Netview*) although they are quite different in the number of edges.

Network Views	Netview	TO Netview	CC Netview
Nodes	150	150	145
Edges	1511	2007	799
Cliques	12	8	8
Clustering Coefficient	0.388	0.37	0.33

Table 5.1: General properties for the three communication network views from Enron network.

5.8 Results

After extracting different partial network views from the enron dataset, as discussed in the previous section, the four different community detection algorithms: Walktrap, Infomap, Label Propagation and Edge Betweenness, which are presented, in Section 5.3 are used to study the level of realism of the resulting communities. We studied how each algorithm performs differently on each view from two perspectives: (a) Topological characteristics of the resulting communities as described in depth in Section 5.8.1, (b) Partitioning quality of each of the algorithm's output as illustrated in Section 5.8.2. Our main focus is to compare the results of the algorithms with each other and not to the unavailable ground truth as highlighted in Section 5.1. We stress on the fact that the community detection algorithms in most cases lead to different results whether it is from the partitioning point of view or the topological characteristics. Hence, we evaluated the resulting communities from both prospectives to complement each other. Our experiments and analysis are implemented using R language and igraph [118] graph library to study and evaluate different community detection algorithms.

5.8.1 Topological Characteristics

This section shows how the four community detection algorithms with the three different network views can impact the topological properties of the resulting communities.

Size, Size Distribution & Singleton: The different community *size* resulted from the various algorithms are highlighted in Table 5.2. While, the *singleton* communities appear in Table 5.3. Figure 5.3 illustrates the *size distribution* for each community (represented in colored bars) across the three different network views with an exception of the singleton communities; where the *x-axis* represents the algorithm and the *y-axis* shows the *size distribution* of communities for these algorithms. For example, Figure 5.3 (a) represents the Netview and shows that WT algorithms has a size distribution for the communities better than other algorithms. It is worth emphasizing the following:

- EB shows the poorest performance across all views specially for the *CC Netview*. This can be observed from the number of communities having more than one vertex (3, 2, and 5 communities in the *Netview*, *TO Netview*, and *CC Netview* respectively) with respect to the total number of the resulted communities, this is a large number of singleton communities for the three views (See Table 5.2 and Table 5.3).

- WT and IM perform reasonably well for both the *TO Netview* resulting in 7 and 8 communities more than one vertex respectively. Similarly for the *CC Netview* which resulted in 10 and 15 communities respectively. Both perform effectively with the *CC Netview* deriving 0 singletons.
- For LP, the communities resulted for the *CC Netview* is relatively good compared to the *Netview* and *TO Netview.* This is illustrated from the ratio between the number of communities that contain more than one vertex (4 communities) with respect to the total number of communities (9 communities) corresponding to $4/9 \approx 0.4$ compared to the 0.75 and 0.67 ratio resulted for the *Netview* and *TO Netview* respectively. For the singletons, interestingly, it extracts a smaller number of singletons than WT for *Netview* and *TO Netview.*

We could claim intuitively that the lower the ratio between the number of communities that contain more than one vertex with respect to the total number of communities, the better distributed the communities and hence, we have a higher chance to derive interesting insights from such communities.

Transitivity: Figure 5.4 shows the transitivity for the communities found by the four algorithms, which show different trends. Transitivity is represented in bars, colors and order of communities are presented in a similar way as in Figure 5.3 so you can quickly observe the impact of size distribution on transitivity. We observed from this figure the following:

- For LP and EB, the larger community size is obtained, the higher transitivity and vice versa. This inverse relation is found to be consistent in all network views.
- It is worth noticing that despite the fact that EB resulted in a poor partitioning (based on the size distribution property), some of the resulting communities score high transitivity (> 0.8).
- For IM and WT, the *TO Netview* is observed to have high transitivity values compared to the *CC Netview* (> 0.5) for most communities. The communities which score transitivity equal to 1 are mainly comprised of 2 or 3 nodes, which follows the intuitive thinking expected from very small communities (i.e. high transitivity can be easily achieved in small communities).

Edge density: As shown in Figure 5.5 which read in a similar way to Figures 5.3 and 5.4, some of the observed interesting points are:

- For LP and EB, a defined relation between edge density and community size: edge density decreases with the increase of community size and vice versa resulting in large sparse communities and dense small size communities.
- For IM and WT, there is difference in edge density values for communities in the *Netview*. This difference decreases in the *TO Netview* and further decreases in the *CC Netview*; communities tend to get close in their edge density values which reflects that they are more homogeneous (all of them range between 0.2 to 0.6).

5.8.2 Evaluation of Partitions

We compare the partitioning extracted from algorithms for each view to check the agreement of the set of nodes in a community. Hence, the agreement corresponds to the proportion of a set of nodes for which two communities agree across each network view. This evaluation is performed using the quality measures discussed in Section 5.5.

Modularity: Modularity is used to measure the quality of partitioning in general, as shown in Table 5.4 to find out which is the best partitioning in each case. We found that:

- IM and WT have the highest modularity when compared to LP and EB. Both IM and WT score very closely modularity index among all views.
- Measured by modularity, the quality of partitioning for all the algorithms is higher in the *CC Netview* than the *TO Netview*.
- Measured by modularity, the quality of partitioning of the *TO Netview* is higher than the *Netview* except for the EB.

Similarity Measures: These are used to compare the resulting partitions to highlight their similarity level. When looking at RI, ARI and NMI in Table 5.5, Table 5.6 and Table 5.7 respectively. We observed that the three similarity measures agree on the order of the similarity values for both the views and the algorithms. However, values of ARI are higher than NMI values which are higher than RI. Because the ordering is the same, our discussion is valid for any of the three measures.

- EB & WT/IM/LP, although EB nearly scores the least modularity for all the network views across the other three algorithms but it shows a high similarity score when compared to them specially for the *Netview*. Generally, these high values do not match with the high divergence in the community sizes distribution. The only view that matches with that divergence is the *CC Netview* and gives low similarity scores as well, where the performances observed to be very low (nearly close to zero) for ARI measure with other algorithms.

Network Views	Netview	TO Netview	CC Netview
IM	7	9	11
WT	4	9	15
LP	4	3	9
EB	21	9	55

Table 5.2: Different community sizes generated from the algorithms: IM, WT, LP and EB for three different communication network views including the singleton communities.

Network Views	Netview	TO Netview	CC Netview
WT	2	2	0
IM	1	1	0
LP	1	1	4
EB	18	7	48

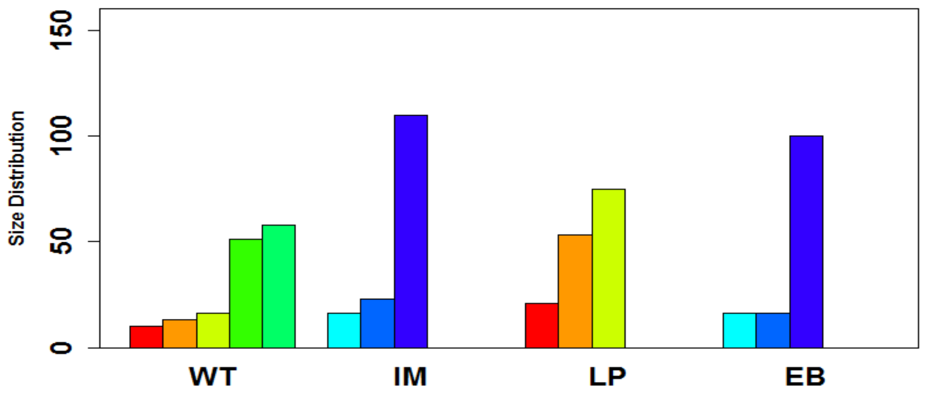
Table 5.3: The number of singleton communities generated from the algorithms: WT, IM, LP and EB for three different communication network views.

- IM & WT score the highest similarity measure when compared to each other for the *CC Netview* followed by the *TO Netview* and that means that their similarity value affected significantly by the type of the relationship in the graph. For example, their values reached 0.9 for the RI measure for the *CC Netview* and this agreed with their corresponding modularity values which is more than 0.5.
- LP & WT obtained the highest values for the *CC Netview* and then the *TO Netview.*
- Most of the algorithms showed better value for the *TO Netview* more than the *Netview* except LP & IM, both of them give higher similarity over the *Netview* than the *TO Netview.*

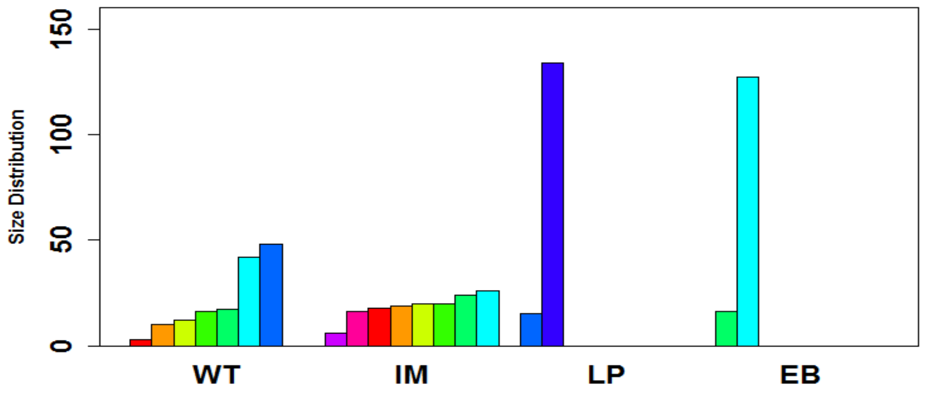
5.9 Discussion

In this section, we will present a discussion based on our study from the previous section that will adopt both views, the topological measures and the evaluation of the performance.

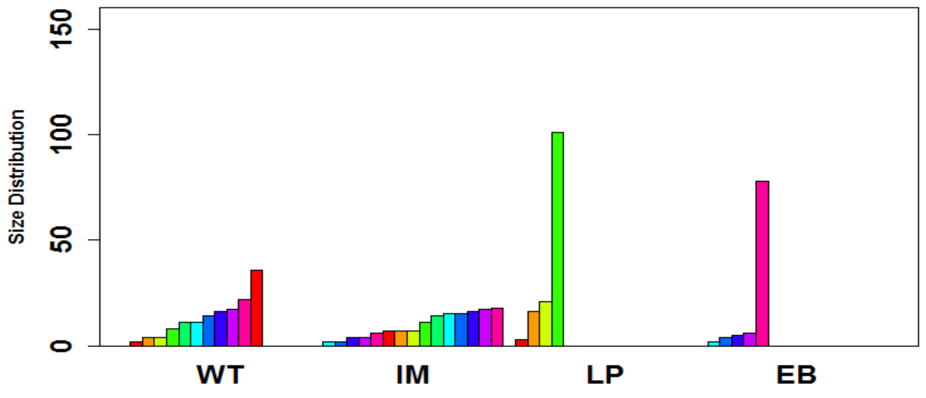
- **From the topological properties point of view:** *size distribution* got affected by both algorithms and the type of the network view. It is clear that changing the views does not impact the rate of *singletons* but the algorithm itself has the main impact on their rate. The *transitivity* and the *edge density* are not always dependent on the size of communities. The partitioning of *CC Netview* with IM and WT worth having an attention as it may give insights that might be close to real-world. Directed networks can be more fertile for partitioning and insightful than undirected ones even if they are denser (e.g. *TO Netview* better than *Netview*). Some directed networks can be more interesting than other directed ones according to the modeled relation type. We



(a) Netview

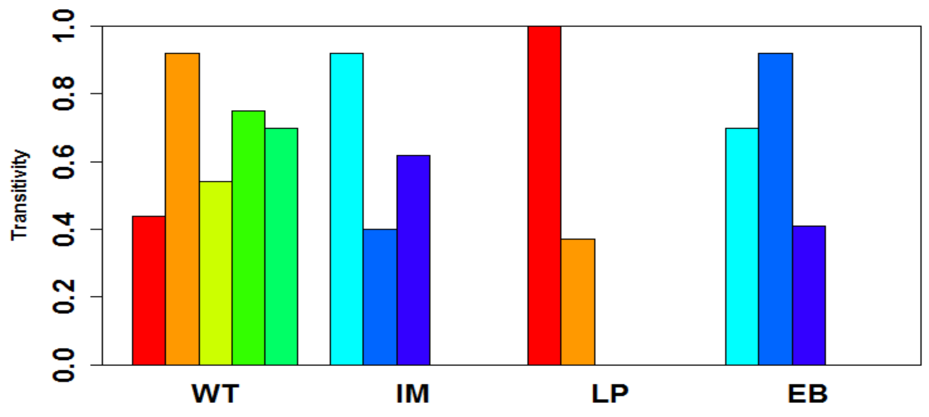


(b) TO Netview

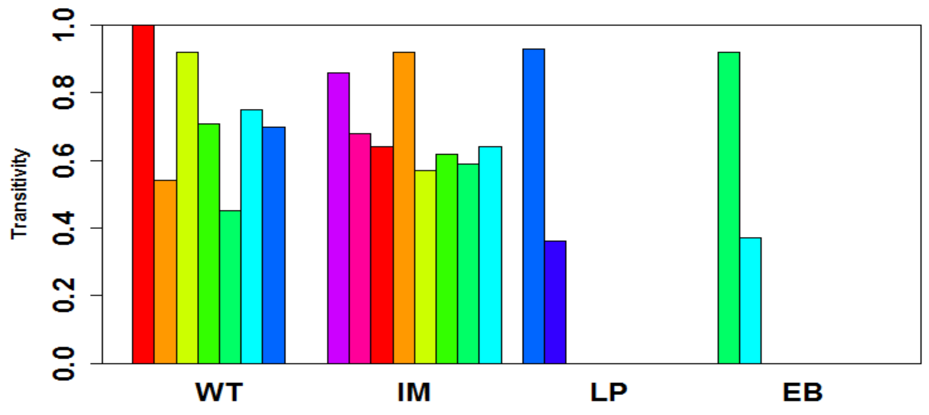


(c) CC Netview

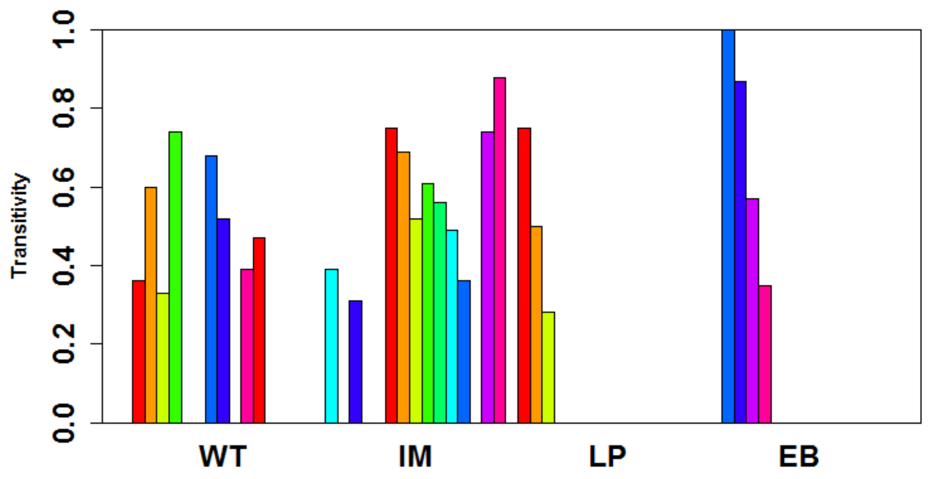
Figure 5.3: Size distribution plotted in bars to represent the measures for each community (*y-axis*) derived from each of the four algorithms: WT, IM, LP and EB (*x-axis*) for *Netview*, *TO Netview* and *CC Netview*. The plotting here considers communities resulting from each algorithm except the singleton communities for 150 employees in Enron network and the colors refer to the unique communites dervied by each algorithm.



(a) Netview

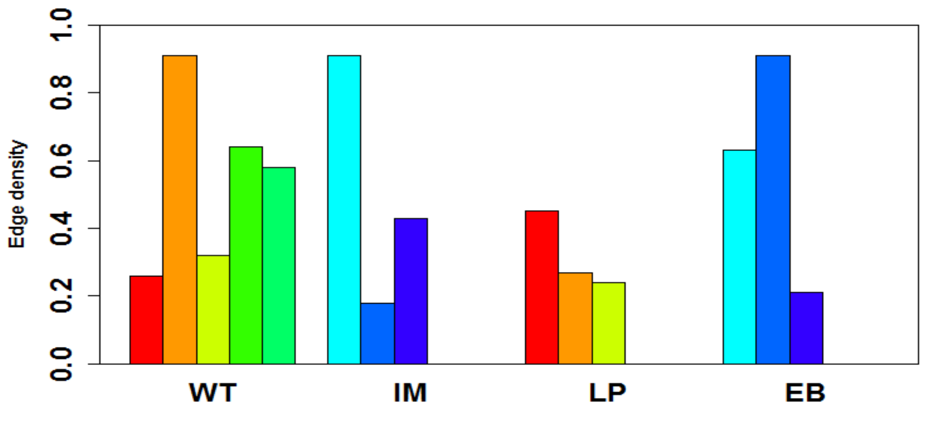


(b) TO Netview

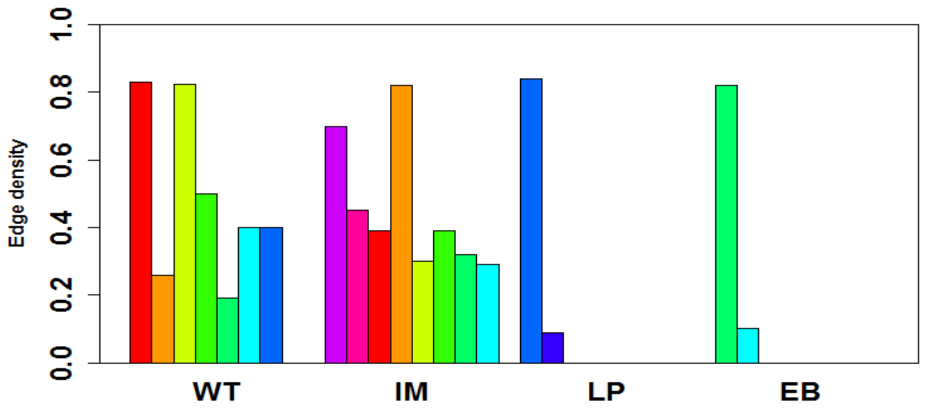


(c) CC Netview

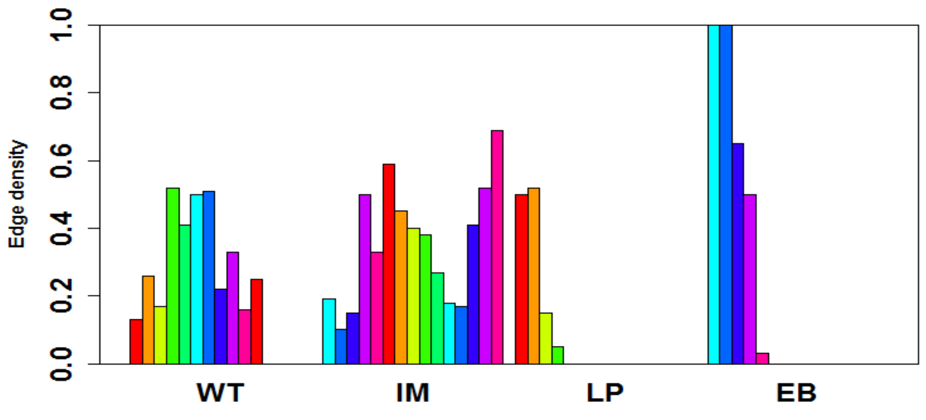
Figure 5.4: Transitivity plotted in bars to represent the measures for each community (*y-axis*) derived from each of the four algorithms: WT, IM, LP and EB (*x-axis*) for *Netview*, *TO Netview* and *CC Netview*. The plotting here considers communities resulting from each algorithm except the singleton communities for 150 employees in Enron network and the colors refer to the unique communites dervied by each algorithm.



(a) Netview



(b) TO Netview



(c) CC Netview

Figure 5.5: Edge Density plotted in bars to represent the measures for each community (*y-axis*) derived from each of the four algorithms: WT, IM, LP and EB (*x-axis*) for *Netview*, *TO Netview* and *CC Netview*. The plotting here considers communities resulting from each algorithm except the singleton communities for 150 employees in Enron network and the colors refer to the unique communites dervied by each algorithm.

can infer this from the *CC Netview* where the derived communities seem to be more appropriate than the *TO Netview* according to our analysis. Interestingly, we found some communities in the views act as a sub-network of those in other views and tend to split into communities in other views. For instance, in WT, some communities in the *Netview* get split in the *TO Netview* and communities in the latter one tends to split in the *CC Netview*. Hence, it is clear that deciding whether the derived communities are well formed or should they be split further into communities, is not an easy task.

- **From the quality of partitioning point of view:** It is clear that among the three network views that WT & IM have the highest scores. The use of different similarity measures did not matter much for our data, this could save time in evaluations. It is observed that the highest scores always achieved for the *CC Netview* and *TO Netview* across most of the measures. This can indicate that: the more relationship is specified in the graph, the higher the quality of the partitions than modeling a graph without specific relationship. It is also clear that the quality results coincide with the topological results, both of them agree that IM is almost as good as WT especially for the *CC Netview* which has been shown to have better topology structure and higher quality of partitioning as well. As the *CC Netview* scores higher results than the *TO Netview* and the latter scores higher results than the *Netview*, therefore, we expect that we can derive more interesting communities when modeling the *BCC* communications.

Generally, we found that the quality of partitioning and the topological structure are equivalent to great extent, however, it is hinted in the literature that they are not always equivalent to each other [95]. We found that the graph based on specific relationship between its nodes in the Enron network play an observable role, it is obvious that modeling a graph based on one type relationship can have an observable impact on optimizing the structure and the quality of communities. Generally, the view can have a big impact, as can the algorithm.

Network Views	Netview	TO Netview	CC Netview
WT	0.35	0.4	0.55
IM	0.23	0.4	0.54
LP	0.11	0.15	0.43
EB	0.2	0.17	0.145

Table 5.4: The quality of partitioning expressed in terms of modularity function for three different communication network views across the four algorithms: WT, IM, LP and EB.

View	**Netview**				**TO Netview**				**CC Netview**			
RI	**IM**	**LP**	**WT**	**EB**	**IM**	**LP**	**WT**	**EB**	**IM**	**LP**	**WT**	**EB**
IM	1.000	0.709	0.712	0.876	1.000	0.319	0.865	0.380	1.000	0.837	0.915	0.704
LP	0.709	1.000	0.423	0.603	0.319	1.000	0.401	0.917	0.837	1.000	0.857	0.636
WT	0.712	0.423	1.000	0.720	0.865	0.401	1.000	0.459	0.915	0.857	1.000	0.681
EB	0.876	0.603	0.720	1.000	0.380	0.917	0.459	1.000	0.704	0.636	0.681	1.000

Table 5.5: *Random Index (RI)* values represented to find the best two matching algorithm for *Netview*, *TO Netview* and *CC Netview* across the four algorithms: IM, LP, WT and EB.

View	**Netview**				**TO Netview**				**CC Netview**			
ARI	**IM**	**LP**	**WT**	**EB**	**IM**	**LP**	**WT**	**EB**	**IM**	**LP**	**WT**	**EB**
IM	1.000	0.355	0.457	0.756	1.000	0.064	0.530	0.072	1.000	0.358	0.550	0.091
LP	0.355	1.000	0.114	0.245	0.064	1.000	0.116	0.770	0.358	1.000	0.492	0.033
WT	0.457	0.114	1.000	0.422	0.530	0.116	1.000	0.143	0.550	0.492	1.000	0.077
EB	0.756	0.245	0.422	1.000	0.072	0.770	0.143	1.000	0.091	0.033	0.077	1.000

Table 5.6: *Adjusted Random Index (ARI)* values represented to find the best two matching algorithm for *Netview*, *TO Netview* and *CC Netview* across the four algorithms: IM, LP, WT and EB.

View	**Netview**				**TO Netview**				**CC Netview**			
NMI	**IM**	**LP**	**WT**	**EB**	**IM**	**LP**	**WT**	**EB**	**IM**	**LP**	**WT**	**EB**
IM	1.000	0.432	0.689	0.701	1.000	0.281	0.739	0.339	1.000	0.642	0.796	0.529
LP	0.432	1.000	0.271	0.285	0.281	1.000	0.323	0.694	0.642	1.000	0.667	0.411
WT	0.689	0.271	1.000	0.629	0.739	0.323	1.000	0.404	0.796	0.667	1.000	0.485
EB	0.701	0.285	0.629	1.000	0.339	0.694	0.404	1.000	0.529	0.411	0.485	1.000

Table 5.7: *Normalized Mutual Information (NMI)* values represented to find the best two matching algorithm for *Netview*, *TO Netview* and *CC Netview* across the four algorithms: IM, LP, WT and EB.

5.10 Related Work

Huge attention has been focused on finding communities in large networks dealing with communities as a set of nodes that have better links across each other compared to the rest of the network. As the question of finding the optimum community detection algorithm is still open and the problem of having a network with a known community structure is a complex problem, many researchers have focused their attention to propose studies and analysis to tackle this area. We will briefly review some of the other work done on comparing different community detection algorithms and on defining communities in an email networks.

Jure et al. [30] explore different community detection algorithms and compare them to understand how each algorithm performs using some quality metrics. They consider community quality as a function of size a finer lens to examine community detection algorithms. Their empirical study conclude that finding the community structure in large networks is a complex problem. Lancichinetti et al. [109] proposed a class of benchmark graphs that create artificial graphs with a known community structure and used them to test popular and common community detection algorithms. The benchmark takes into account the heterogeneity of both community size and the degree. Orman et al. [95]

provide a comprehensive study on community detection methods to evaluate and asses the derived communities with the reference structure using an artificial network derived by LFR model [109] that proposed by Lancichinetti. Qian et al. Lancichinetti et al. [112], studied the characteristics of communities of some types of different complex networks. They show that although different methods derive different clusterings. However, for similar classes of networks, the statistical properties of their communities are quite similar.

Qian et al. [119] developed an algorithm based on link mining to find the community structure within the Enron email corpus. Leskovec et al. [30] studied a number of different real email networks, including some email networks of large organization and the Enron email network to empirically compare two different clustering methods. Guimera et al. [111] conducted a study of community structure of email networks using emails in the university. Moradi et al. [120] evaluate the community detection algorithms using email networks to separate spam from legitimate email through clustering them into unique clusters. Nawaz et al. [121] proposed a community detection method to find the groups of email users according to their structural intimacy, the derived communities are evaluated using different quality measures.

5.11 Summary

The aim of this chapter was to study the derived communities based on different relationships/interactions representation for the Enron email network using four different community detection algorithms: Walktrap, Infomap, Label propagation and Edge Betweenness. The goal from this study is to discover how the topological properties and the quality of partitioning of communities got affected by two factors: various network interactions and multiple community detection approaches. We studied the idea of selecting an interesting interaction and a reliable algorithm for detecting communities which has been a challenging task with the lack of ground truth.

We studied some topological properties of the estimated communities that resulted from each algorithm applied on three network views. We then assessed the results through evaluating the quality in terms of comparing partitions using modularity measure and other similarity measures. The considered measures mostly agree with each other on the quality of partitioning with small differences and agreed to a great extent with the topological properties.

Chapter 6

An Ensemble approach for Detecting Communities in an enterprise network

In this chapter, we propose an ensemble community detection approach for social networks called **Multi-criteria Community Fusion (MCF)**. MCF is based on combining multiple community detection algorithms with the aim of promoting the derived communities and trying to increase the quality of the derived communities from using just one algorithm. We will explore the behavior of the ensemble approach on a real-world network and study its performance relative to other traditional existing approaches.

6.1 Motivation

In the recent years, many studies have focused their attention on the community detection problem in social networks. As discussed previously in chapter 5, most of the available algorithms target a single function (the objective function) to optimize the derived communities. For example, these functions might focus only on modularity, betweenness centrality or edge density. In other areas of machine learning, fusing or ensemble methods of clustering or classification have been common [122]. It has been proved that results can be improved when the data is fused from several data sources [123]. Following the same approach applied to clustering methods, we aggregate information from different community detection algorithms based on different objective functions to propose a new approach for deriving communities based on hybrid fused communities either from different algorithms or different graph views. We name this approach as **Multi-criteria Community Fusion (MCF)**.

6.2 Introduction

Several studies deal with community detection problem as an optimization problem [124]. Lots of these studies were based on either collaborative algorithms or evolutionary methods to handle the optimization problem [125]. Community detection algorithms are usually based on one criteria and hence their optimization target is based on specific objective. A network can be split into different communities due to various community definitions. With the presence of several outputs, there is a need for a criteria to rank partitions to be able to discard some and therefore a better option could be combining multiple outputs into a new partition. Exploitation of information from different partitions is considered very important for detecting communities especially in dynamic systems [126] [127] with the increase of time-stamped datasets [128]. Another problem, which we discussed in previous chapter, is that most of these algorithms require a ground truth to help in verifying this optimization process. In order to handle this problem, we deal with community detection algorithms from an ensemble perspective. We introduce an ensemble approach based on multiple criteria that help when no defined communities are known. As it is not always easy to asses how well the structure of the detected communities matches the actual existing one, our difficulty is the absence of a ground truth for the structure of communities. So we are trying to get and explore this proposed ensemble definition for communities, aiming to assess its effectiveness. Thus, we target the impact of this hybrid fused approach on user community structure.

Roughly, our assumption is that if two users exist in the same community generated from different algorithms then there is a higher probability that these users exist in one community in real world. Clustering these users after figuering out how often they exist in single communities from different approaches can result in clustered communities based on the similarity in their connection strength.

The approach is tested on a dataset that represents a real enterprise network within **IBM** corporate called **IBM Connections**[1] to generate a comprehensive hybrid fused communities based on multiple criteria. In this chapter, we introduce our MCF approach, we will investigate if the approach has potential to be an effective approach for identifying complex network structures through fusing diverse community detection approaches. The aim is try to capture better and more accurate communities using multiple objective functions and to explore if this ensemble approach is revealing a concrete community structure better than using a single algorithm. To the best of our knowledge, no work in the community detection area has been devoted to either ensemble communities on multiple objective functions or on a real-world business network (*IBM Connections* in our case). However, there are some similar approaches that merge different community structures but using different methodologies and synthetic/artificial network (e.g. [107]).

[1]https://www.ibm.com/us-en/marketplace/enterprise-social-collaboration

6.3 Research Settings and Methodology

In this section, we provide an overview of the dataset used in this chapter, *IBM Connections* dataset. Then, we will present our procedures for preprocessing and preparing the data prior to running the experiments.

6.3.1 IBM Dataset Overview

IBM Connections is an enterprise social network platform that provides a rich source of information about collaborations across several applications and interactions between employees. It is the internal business network within IBM that enables employees communicate through different types of social applications: **Blogs** enable a user to write on someone's wall, comment and like posts of others; **Forums** allow users to create discussions about new topics and enable them to post; **File sharing**, allows users to upload files and others can share or download these files; **Wikis** enable users to edit a wiki created by someone; **Communities** allow the inclusion of different interests from the topic or project perspective, an employee can follow or be a member in a community according to his interests in certain topic or according to his involvement in specific project. Each application allows actions such as commenting, liking and mentioning these actions reflect the level of engagement of employees either with each other or to the content itself. Content items can include hashtags, for example, where hashtags indicate topics discussed in a microblog post.

As for interpersonal interactions between employees, *IBM Connections* allow employees to friend each other (or connect reciprocally), follow each other (someone's activity), tag each other through descriptive tags and invite people to other's network. Employees are notified by these actions through email notifications. Generally, the data includes content, dates, activities, people who created or involved in any action within the network was originally collected using an API.

6.3.2 Data Preparation

The analysis of this chapter was performed over selective data from *IBM Connections* due to some limitations in accessing the data for privacy and ethical issues. However, this work is focused on certain activities and information between employees. Python scripts were implemented to anonimize the data before performing our analysis, especially for the existing personal IDs of employees, and removing any personal or sensitive information related to the employees from the data.

6.3.3 Methodology

As shown in Figure 6.1, our experiments involve different stages prior to applying the MCF approach: first, the methodology in dealing with the data involves creating four different network graphs according to different semantic indicators explained in Table 6.1, each reflects a selected social activity or interactions between employees and distinguishing their interpersonal relationship within *IBM Connections*. The first graph is a **Friendship graph**, this reflects the friendship network for employees. The second is an **Unweighted Tagging graph**, this reflects all tagging actions between employees, relation between employees represents who is tagged by who in the network. The third is a **Weighted Tagging graph**, this reflects the same tagging actions like the previous but in addition considering the frequency of tagging using weights. The fourth is a **Managing graph**, this reflects the hierarchical structure between employees, relation between employees represents who is managed by who in the organization. The graphs represent people working within IBM in Ireland.

Application	**Indicator Semantics**
Friends	Person friends others (reciprocated)
Tagging	Person tag others and tag themselves
Hierarichy	Person managed by another
Community	Person follows a community
	Person member of a community

Table 6.1: Examined Indicators in our study from *IBM Connections*

After creating the graphs, we apply different community detection algorithms from the existing state-of-the-art: **Infomap** (IM), **Walktrap** (WT), **Label Propagation** (LP) and **Louvain** (LV) that are reviewed in chapter 5. The four algorithms will be used as an input for the proposed MCF approach.

6.3.4 Concept behind the Proposed Approach (MCF)

The approach is based on the following assumption: if two users (employees) exist in the same community in most of the community detection algorithms then they are more likely to have similar interests and consequently have a higher probability to exist in one community in the real world. This approach aims to increase the strength or the degree of connectedness by placing similar users in one community.

The input is a network **G** and different community detection algorithms A, B, C, D. Following Figure 6.2, the output is the hybrid fused communities. The following steps summarize the MCF procedures:

1. Apply **N** community detection algorithms say A, B, C and D and a clustering method K on graph **G** which will yield to different partitions P_A, P_B, P_C and P_D respectively.

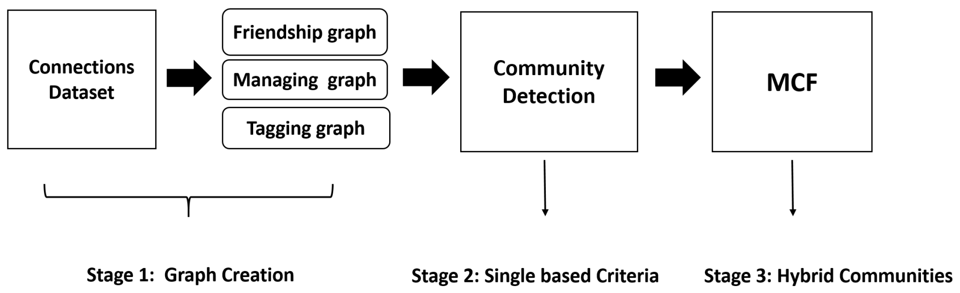


Figure 6.1: Stages of ensemble Community Approach

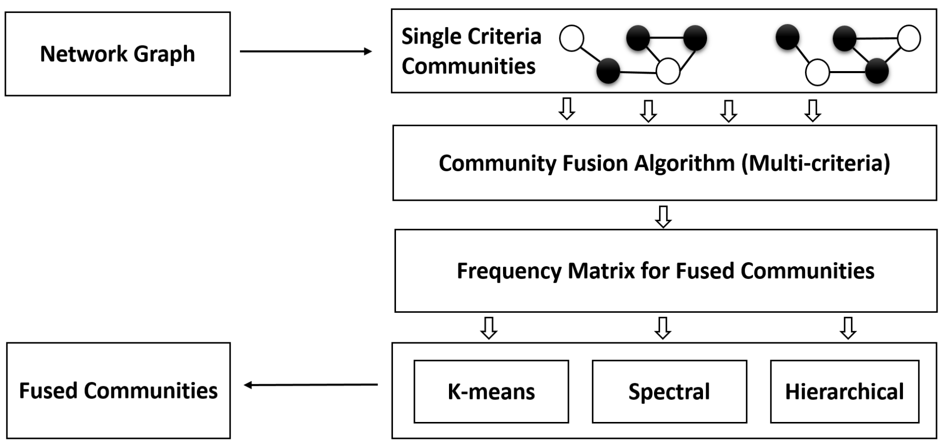

Network Graph
Single Criteria Communities
Community Fusion Algorithm (Multi-criteria)
Frequency Matrix for Fused Communities
K-means
Spectral
Hierarchical
Fused Communities

Figure 6.2: Multi-criteria Community Fusion (MCF) Approach

2. Compute the frequency matrix $\mathbf{M}$ such that M_{ij} is the number of partitions where user i and user j appear in the same group/community across partitions P_A, P_B, P_C and P_D .

3. Calculates the silhouette score (see 6.4.2 for details) for the frequency matrix derived in step 2 to find the optimum number of cluster $\mathbf{S}$.

4. Apply $\mathbf{K}$ clustering method to cluster users.

5. Return the members for each community based on silhouette score for the optimal clustering.

6. Generate the hybrid fused communities based on clustering users.

The MCF approach is implemented using Python language and machine learning libraries for clustering like scikit-learn[2]. R language and graph analysis libraries (e.g. igraph) are used to generate graphs and to experiment community detection approaches.

[2]http://scikit-learn.org/stable/

6.4 MCF Approach

In this section, we describe in depth the proposed approach, which includes the formulation of an ensemble representation of community detection algorithms and then converting the problem of community detection to data clustering.

6.4.1 Aggregating Communities

After generating different relation-based graphs from the *IBM Connections* dataset, we applied four community detection algorithms based on different objective functions (discussed in chapter 5 on each of the generated graphs. Each algorithm produces a possibly unique partition. After generating multiple different partitions, we aggregate all of these partitions in a matrix form which is denoted as **Frequency Matrix**:

$$M_{frequency} = \begin{array}{c|cccc} \text{users} & A_1 & A_2 & \dots & A_n \\ \hline A_1 & m_{1,1} & m_{1,2} & \dots & m_{1,n} \\ A_2 & m_{2,1} & m_{2,2} & \dots & m_{2,n} \\ A_3 & m_{2,1} & m_{2,2} & \dots & m_{3,n} \\ \dots & \dots & \dots & \dots & \dots \\ A_n & m_{n,1} & m_{n,2} & \dots & m_{n,n} \end{array}$$

This Frequency Matrix measures the frequency of the co-occurrence of existing each pair of users in the same community. The Frequency Matrix $\mathbf{M}$ is an $n \times n$ matrix, in which M_{ij} indicates the number of times in which user i and user j assigned to the same community divided by the number of partitions (or algorithms) as shown in Figure 6.3. The goal of this matrix is to include a summary of all the information produced by different community detection algorithms which corresponds to the number of times any two users appeared in the same community. The matrix looks like the adjacency matrix if we represented the network in a matrix and not a graph, however, it is much denser as there is an existing edge whenever two nodes appear in the same community (at least once).

It is worth emphasizing that we intended not to specify a certain threshold for the value in the Frequency Matrix and include all the values to take in consideration the percentage of error (noise) that is delivered by each approach.

6.4.2 Determining the Number of Clusters

Determining the optimum number of clusters is usually a prior step before applying clustering process where the resulting clusters may differ when applying different number of

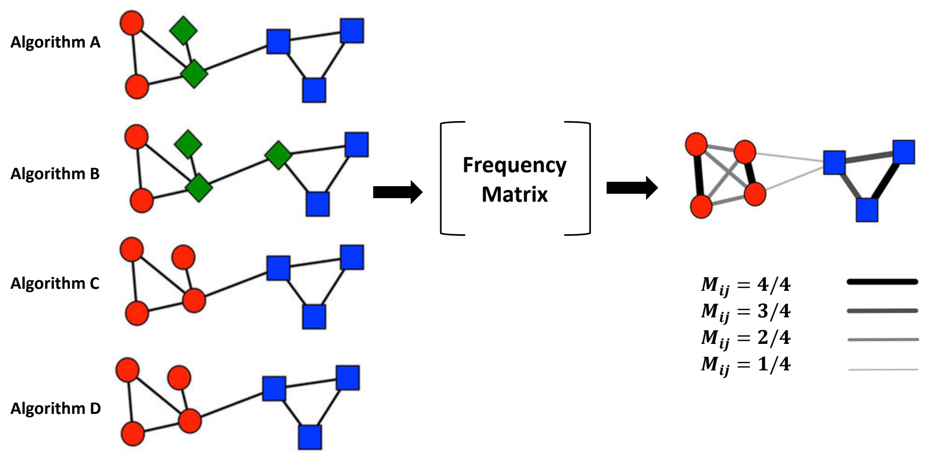


Figure 6.3: Community aggregation to create the Frequency Matrix.

clusters. In terms of selecting the number of clusters, different methods have been utilized to get an estimate for the correct number of clusters. **Silhouette Coefficient** or **Silhouette Index** is proposed by Kaufman et al. [129]. This measure is used as a metric to find the number of clusters which maximizes the silhouette coefficient to identify the quality of the clustering results. Silhouette index range between -1 and 1, where -1 corresponds to poor clustering while $+1$ corresponds to a highly dense clustering. A high silhouette score indicates a highly dense clusters internally and well separated from other clusters externally which agree with the methodology and concept of clustering. The higher the score, the more appropriate the results from clustering. Silhouette is defined as:

$$Silhouette = \frac{b - a}{\max(a, b)} \tag{6.1}$$

where a is the mean distance between a sample point (an employee in our case) and the rest of the points in the cluster and b is the mean distance between a sample point and all the points in the nearest cluster. The silhouette coefficient used in our work is based on measuring the Euclidean distance [130] between clusters to evaluate the quality of the users' clusters that are derived from the MCF approach. We identified the optimum number of clusters for each graph type. This is achieved by applying the clustering methods and varying the number of clusters on each graph and generating the corresponding silhouette coefficient in each case.

6.4.3 Applying Clustering Methods

The traditional definition for clustering is that objects that are assigned to the same cluster are more similar to each other than the objects in other clusters [131]. There are numerous definitions for the similarity or even the dissimilarity measures. Measuring the degree of similarity/dissimilarity can be through different distance measures like: Euclidean [130] or Manhattan distance [130]. We consider three different clustering methods: Hierarchical, K-means, and Spectral. It worth notice that to the best of our knowledge that this the first work to compare different clustering within the ensemble concept.

- **K-means clustering** [132] is a widely used method which assumes the number of clusters k is known and is an iterative approach which tracks the cluster means. It tries to cluster data based on their similarity by trying to find patterns. At first, it assigns each observation randomly to a cluster, and then it tries to find the centroid for each cluster. The approach keeps iterating through two steps: 1) Reassigning the data points to the cluster whose centroid is closest, 2) Calculating new centroid of each cluster. Both steps are repeated until the sum of the Euclidean distance between the data points and their respective cluster centroids cannot be reduced further.

- **Spectral clustering** [133] is a popular method which includes recursively splitting the graph into subgraphs using various criteria. It discovers clusters based on the eigenvector of a matrix that is related to the adjacency matrix with the aim of finding a good cut of the graph into subgraphs. The process is repeated until K clusters are found. It can be considered as a “pre-processing” step to change the feature representation before passing the new representation to K-means.

- **Agglomerative Hierarchical clustering** [134] considers each item is in its own cluster then it starts to merge nearest clusters based on a linkage function and the process is repeated until only one cluster is left. This produce a sequence of clustering assignment which can be visualized in dendogram or tree (a diagram that displays the hierarchical sequence of clustering assignments). The most common linkage functions are: *single linkage* which measures the least dissimilar pair between groups, *average linkage* which measures the average dissimilarity across all pairs and *complete linkage* which measures the most dissimilar pair between groups. The choice of linkage defines the way of measuring dissimilarity between groups of points.

Note that the clustering methods used in this work focus on clustering the output of the hybrid fused communities, with the hope that clustering can provide a higher quality communities. We will explore the impact of using these different clustering methods in the MCF approach.

6.5 Exploratory Analysis

In this section, we will explore the different variables we include in our experiments for testing the MCF approach.

- *Exploring different Networks Interactions:*

  Each of the created graph based on different interactions and relations have a unique structure and size reflected in the number of nodes and edges according to the represented relation as shown in Table 6.2. The difference in the number of nodes reflects the number of the employees who contributed and reacted within *IBM Connections* network. However in the Managing graph, there is a higher number of nodes as it corresponds to the existing hierarchy of the organization and not corresponding to human activities.

	Friendship Graph	**Tagging Graph**			**Managing Graph**
		Unweighted (with self-loops)	**Unweighted (without self-loops)**	**Weighted**	
Nodes	3652	2548	2548	2548	5465
Edges	33654	40365	20635	4392	5448

Table 6.2: Overview about the structure of the extracted relation-based graphs from *IBM Connections*

- *Exploring different Clusterings Methods:*

  We experimented MCF for the Friendship graph using different clustering methods. As shown in Table 6.3, we used three clustering methods: K-means, Hierarchical and Spectral clustering to explore whether the type of data clustering applied will affect the defined number of delivered communities by the ensemble approach or not. The resulting clusters correspond to the fused communities found in the network. We observed that each clustering method derived a different number of clusters. In order to obtain a possibly optimum number of clusters, we ran the experiments 60 times until the optimum number of clusters having the highest silhouette index is achieved and the number of clusters no longer changed. The Friendship graph is the graph that derives the largest number of clusters, it derives for K-means, Hierarchical and Spectral, 45, 44 and 52 clusters respectively with relatively high values for silhouette index. The clustering methods generated only 2 clusters for each of the Unweighted Tagging, Weighted Tagging and the Managing graphs.

	K-means		**Hierarchical**		**Spectral**	
	Clusters	Silhouette	Clusters	Silhouette	Clusters	Silhouette
Friendship Graph	45	0.74	44	0.73	52	0.73
Unweighted Tagging Graph	2	0.85	2	0.84	2	0.84
Weighted Tagging Graph	2	0.85	2	0.84	2	0.84
Managing Graph	2	0.56	2	0.59	2	0.59

Table 6.3: Clustering results across multiple relation-based graphs derived from *IBM Connections using K-means, Hierarchical and Spectral clustering.*

- *Exploring different Objective Functions:*

  Referring to chapter 3, *Modularity* is one of the most commonly used methods to evaluate the quality of partitioning a network into communities. We used it to explore the quality of the communities detected within the tested network as it is a common measure used to determine and compare the performance of community detection algorithms as shown in Table 6.4. Interestingly, we found that all algorithms performed strongly with the Managing graph with modularity score more than 0.9. The Friendship graph came in the second place for all algorithms with modularity more than 0.6, followed by the Weighted Tagging then the Unweighted Tagging graph. We will discuss and validate these results later in our experiments.

	LP (Neighborhood)	**IM (Compression)**	**LV (Modularity)**	**WT (Random walks)**
Friendship Graph	0.62	0.60	0.66	0.63
Unweighted Tagging Graph	0.49	0.12	0.58	0.43
Weighted Tagging Graph	0.52	0.55	0.61	0.55
Managing Graph	0.92	0.92	0.97	0.90

Table 6.4: Performance of algorithms: LP, IM, LV and WT according to *Modularity* using different objective functions.

6.6 Evaluation Criteria

In this section, we define our criteria for evaluation and measuring the quality/performance of the MCF approach relative to existing community detection algorithms on different graphs to discover if the approach will result in robust communities using *IBM Connections* than using single approach or if there are other factors that impact the quality of the ensemble approach. Generally, there are two kinds of measures that can be applied to measure the performance of communities: the first is used to measure the agreement of the resulted communities with known labeled ones and the second is used to measure the goodness of the community structure without any previous information. Obviously, the second type will be more suitable for assessing the MCF approach as the ground truth is not available. We will discuss some of the measures that have been proved to achieve best results for deriving communities in the literature.

6.6.1 Surprise Score

A *Surprise (S)* metric is a global measure which is used to evaluate the quality of a partition [135] and to improve the knowledge of how the approach performs. It was found to have an efficient behavior across many networks [135], [136]. The higher the Surprise score, the more maximally connected the communities internally (intra-links) and the maximally isolated from each other (inter-links). The Surprise score is calculated using the following formula:

$$S = -\log \sum_{j=W}^{\min(Z,n)} \frac{\binom{Z}{j}\binom{X-Z}{n-j}}{\binom{X}{n}} \tag{6.2}$$

The measure computes the probability of partitioning a given communities from a certain network. For a network with n links, X is the maximum number of possible links, Z is the possible maximum number of intra-links for a given partition and W is the total number of intra-links in that partition.

6.6.2 Variation of Information

The *Variation of Information (VI)* is based on a distance measure which indicates the level of difference between two partitions (or two clusters) taking into account the structure and the features that are common. It measures the amount of information lost and gained in changing from partitioning P to partitioning P'. This metric is proposed in [137].

According to the literature [135], it has been shown that Surprise can evaluate the efficiency of partitions effectively while *Modularity* cannot. This is according to the formal definition of *Modularity* which is based on achieving high density of links however the mathematical formula on which *Modularity* is not based on taking in consideration the number of nodes which achieve the high density of links. In addition, *Surprise* can detect small communities which can not be detected by *Modularity.* Based on this, we will not use *Modularity* as a performance measure to assess our approach and will focus on *Surprise* and *VI* measures. The ensemble approach is evaluated and compared across the other four techniques using the *VI* and *Surprise* methods on the four relationship-based graphs. If maximizing *Surprise* is the optimum methodology for defining communities due to its correlation with what *VI* suggests, it should be the possible way to choose the best approach which has maximum *Surprise* value in each particular network.

6.7 Results

We believe that the fusion of different algorithms can lead to more accurate communities than the ones derived from one approach. Our approach depends on selecting multiple algorithms, each is based on different approach for delivering communities and thus aiming to achieve better solutions comparing them over various experimental variations.

To better explore the effectiveness of this ensemble approach, we included and studied different variations: 1) the different communities derived from different community detection algorithms, where each algorithm is based on different objective function; 2) the different types of interactions or relationship across employees within *IBM Connections* dataset; 3) the effect of edge weights on algorithm performance; 4) testing different clustering methods on the hybrid fused communities.

Our experiments address these variations and their impact on the MCF approach based on selection of evaluation criteria, which include measuring the performance of the MCF approach relative to other state-of-the-art algorithms by measuring the Surprise score. In addition to comparing the level of agreement on the delivered communities between MCF and other algorithms using the Variation of Information (VI) measure.

- *Testing the impact different interactions on the delivered communities for both MCF and other state-of-the-art algorithms:*

  Referring to Figure 6.4, for the defined communities in the Friendship graph, both the ensemble MCF and the four individual algorithms (WT, IM, LP and LV) agree that the highest Surprise score is obtained. Interestingly the highest Surprise score is obtained by the IM followed by the MCF. While WT, LP and LV algorithms delivered a lower Surprise score than MCF. Again, all algorithms and MCF agree that the Unweighted Tagging graph comes in the second place with lower Surprise score than the Friendship graph. However for this graph, individual algorithms behaves better than the ensemble MCF, except for the IM. We observed that MCF and IM behave similarly with a very tiny difference in their Surprise values for both the Friendship and Unweighted Tagging graph. For the Managing graph, the interesting observation is that all algorithms are in same range of Surprise except MCF which behave poorly relative to others. This matches the hierarchical nature that imply the increase of inter-community links within the graph which consequently leads to weakness in the community structure. For the Unweighted Tagging graph, both the individual algorithms and the ensemble MCF approach behave poorly which means that weighting the edges might have a bad impact on the quality of the delivered communities.

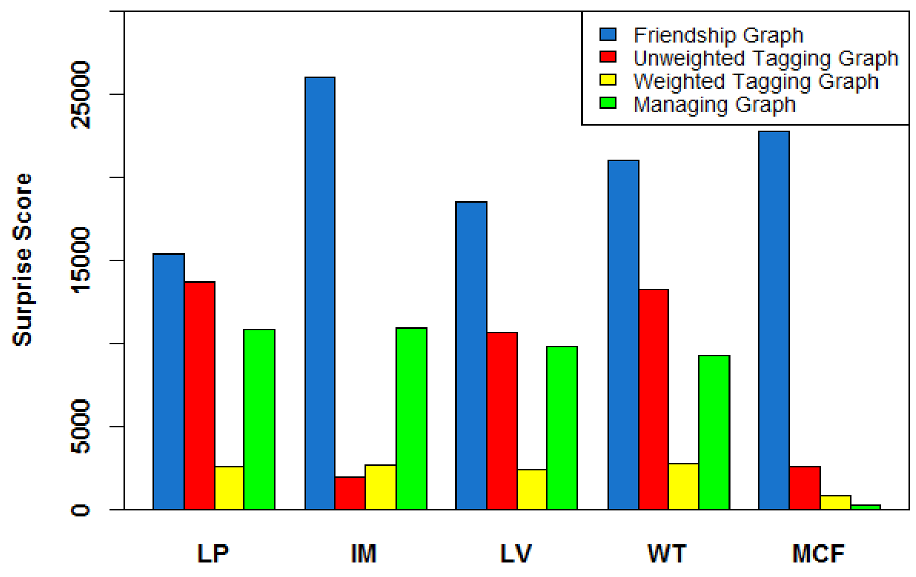


Figure 6.4: Performance of algorithms: LP, IM, LV and WT according to *Surprise* Score compared to MCF using K-means clustering.

In order to test our analysis for the Surprise scores, we visualized using Gephi[3] tool some of the networks to reflect the real structure. Colors and size of nodes represent employees that are having many connections (the bigger the size of the node higher the degree of that node). According to Figure 6.5, we found the Friendship graph has the potential to have interesting communities due to its dense network structure which reflect good network connections between employees. While for the Tagging graph Figure 6.6, visualization obviously show useful information about communities and that is because most of the employees within the network tag themselves instead of tagging others. We observe that even if there are some employees tag others, there is no reciprocal relation of tagging neither between the employees nor their neighbors. We found it not useful to visualize the Managing graph due to the sparsity found in the structure of the network but still this can justify the poor performance appear in the experiments which corresponds to the low Surprise value for MCF approach.

- *Testing the difference between multiple objective functions and the ensemble one on different network interactions:*

To measure the closeness of MCF results to other algorithms, we measure the VI scores across each individual algorithm and the MCF approach. Figure 6.7 shows

[3]https://gephi.org/

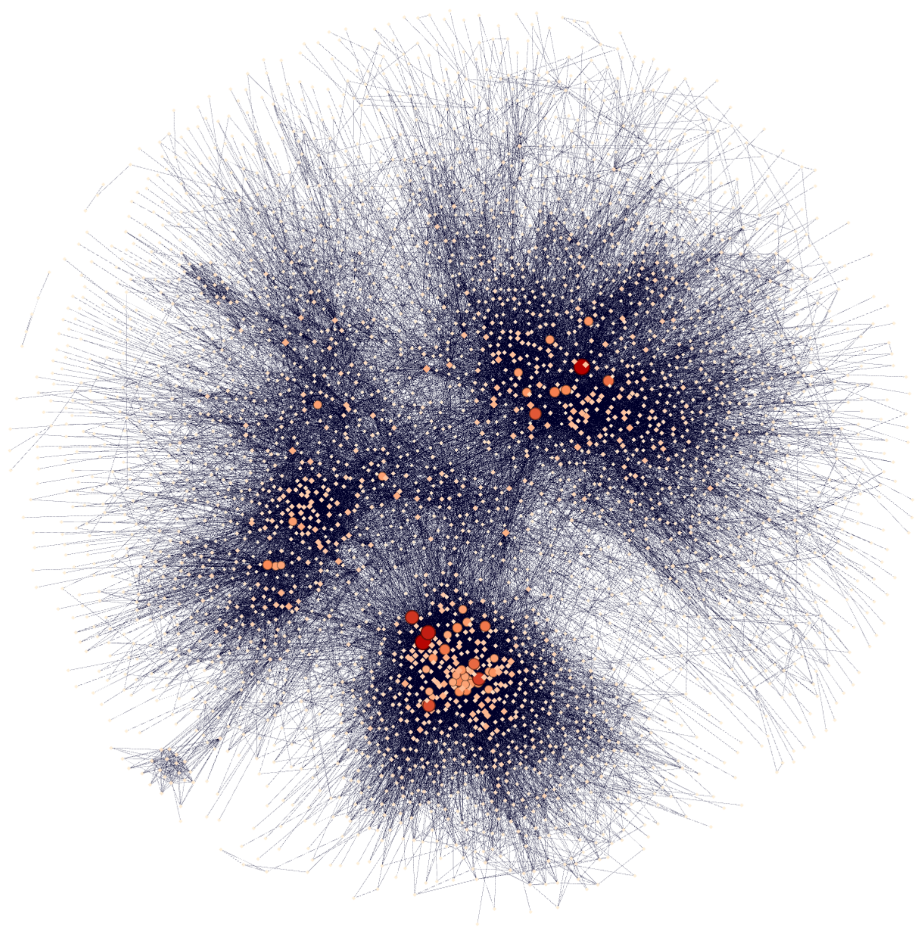


Figure 6.5: Friendship network for employees in Ireland within *IBM Connections*. Deep color and big size of nodes represent employees that are having many connections.

the congruence between the communities delivered by each of the four algorithms and the ones delivered by the MCF approach. For the Friendship graph, MCF and other algorithms lead to deriving the lowest values 1.67, 1.75 when compared to WT and IM respectively. While for Tagging and Managing graphs, high VI are achieved. Generally, the delivered communities within the Friendship graph impacted positively the VI and the Surprise scores.

- *Testing the impact of different clustering methods on the delivered communities by MCF:*

  In order to evaluate the MCF with each clustering method, we re-calculated Surprise and VI measures for MCF based on each clustering method for the Friendship graph as shown in Figure 6.8. A slight difference is obtained between the Surprise values for the MCF across the three clustering methods. However from the ranking perspective, Spectral clustering come in the first place followed by K-means clustering. However,

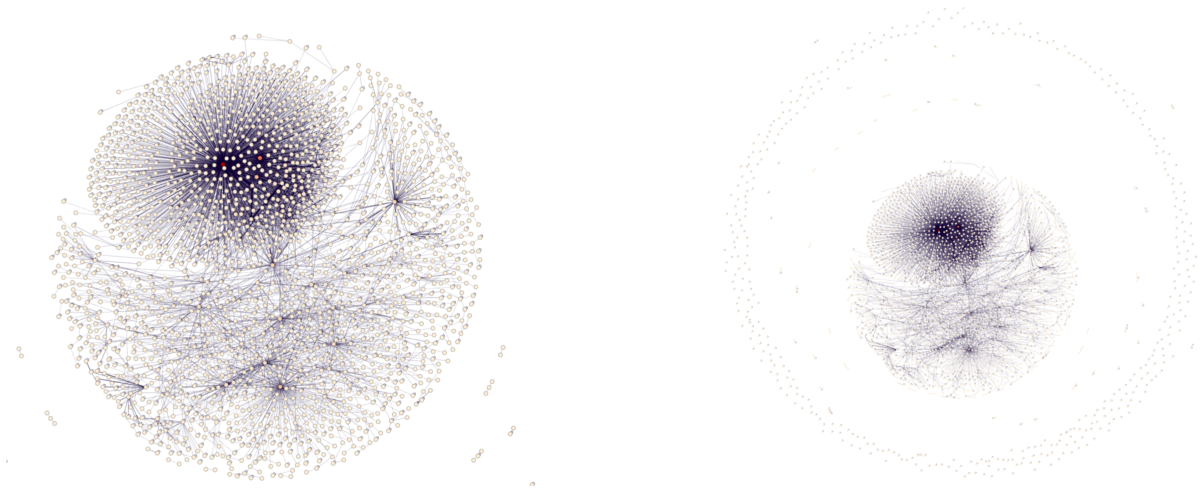


Figure 6.6: Tagging network for employees in Ireland within *IBM Connections*, the figure on the right side includes employees who tag themselves and employees who tag others. The figure on the left side focuses on employees who tag each other. Deep color of nodes represent employees that are having many connections.

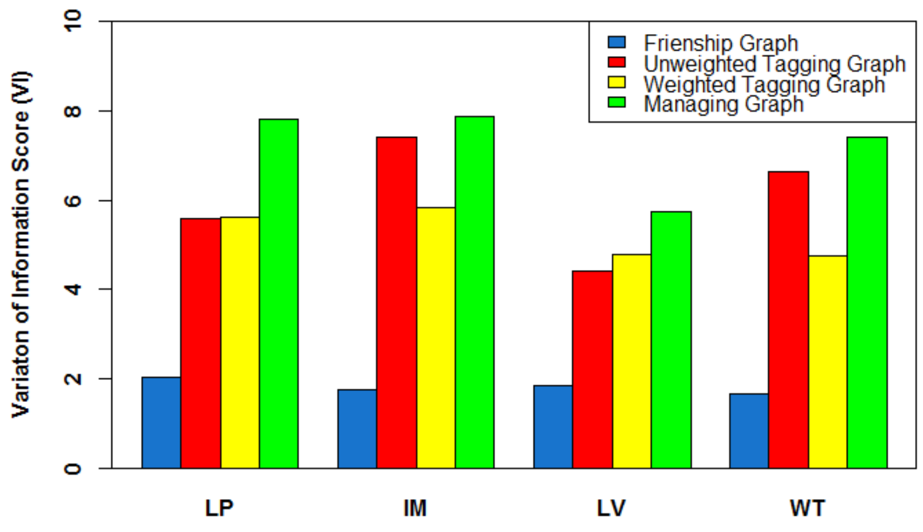


Figure 6.7: Performance of algorithms: LP, IM, LV and WT according to *Variation of Information (VI)* Score compared to MCF using K-means clustering.

the three clustering methods do not affect the rank of the MCF across other state-of-the-art algorithms maintaining IM in the first place and MCF in the second place.

- *Testing the difference between multiple objective functions and the ensemble one using different clustering methods:*

  Figure 6.9 compares the level of variation in the defined communities in the Friendship graph for MCF using three clustering methods and other algorithms. A high VI score obtained between LP and MCF using the three clustering methods. MCF using K-means clustering showed low VI values with WT, LV and LP. On the other hand,

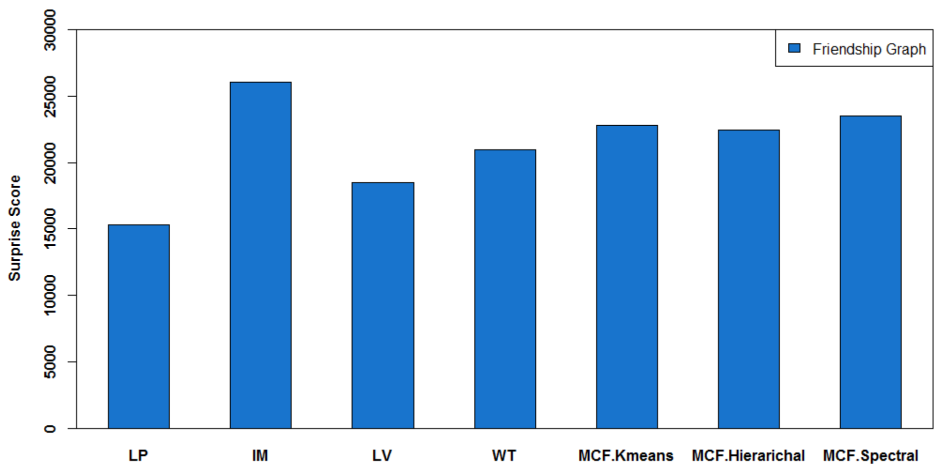


Figure 6.8: Performance of algorithms: LP, IM, LV and WT according to *Surprise* compared to MCF obtained using K-means, Spectral and Hierarchical clustering for the Friendship graph.

MCF using Spectral clustering showed the lowest VI value with IM.

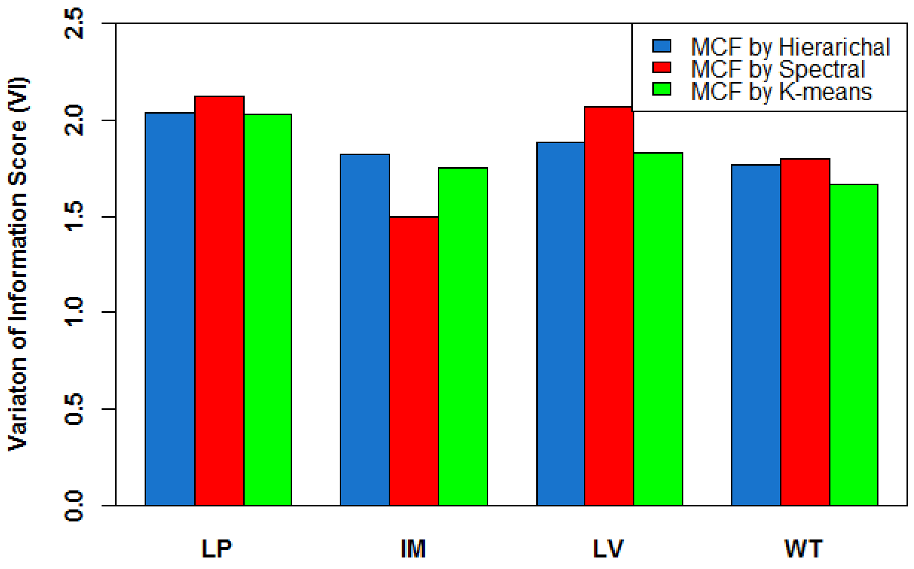


Figure 6.9: Performance of algorithms: LP, IM, LV and WT according to *Variation of Information (VI)* compared to MCF obtained using K-means, Spectral and Hierarchical clustering for the Friendship graph.

In summary, the results demonstrate that not always choosing an ensemble approach does not always results in better performance than the tested state-of-the-art algorithms. As for the complexity of the MCF approach, it will depend on the used clustering method. For example, for Hierarchical method, the complexity will be $O(n^2)$, therefore the MCF will have the same complexity of K-means.

6.8 Experimental Checks

Various experiments are explored for generating ground truth within IBM dataset which aim to more validation for our results. By using the MCF approach, we can get some sense of the certainty we have of an employee belonging to a particular community. In order to try to validate this, we conducted some extra tests in the hope of comparing real existing communities within *IBM Connections* dataset. The first test includes checking employees who follow or are a member of certain groups. The second test includes checking employees who not only follow or are a member in same group but also have a friend relationship. Neither of these checks seemed to reveal useful information about communities, but it did show that following or being a member of a group of certain interests in *IBM Connections* does not give solid information for the definition of communities. In order to confirm this conclusion, we applied Jaccard similarity [138] between each pair of employees to find how many communities they have in common relative to the total number of communities they follow or being a member in given the that they are already friends. We hoped that these scores can reflect the similarity in interests between the employees. Although the Jaccard scores were very low but they agree with our conclusion about the definition of groups within the network.

6.9 Discussion

Our contribution in this chapter focuses on investigating the level of community enhancement by proposing an ensemble approach MCF on a real enterprise network. Studying different variables in the experiment allow useful variations to test the performance of MCF approach. To the best of our knowledge, this is the first time to propose an ensemble approach based on multiple optimization function for community, studies different kinds of interpersonal interactions within a real enterprise network and not an artificial network that can possibly reveals useful insights about the community structure within large enterprise companies (such as IBM). Besides, experimenting with multiple clustering methods and investigating if the ensemble concept inherits the high performance resulting from the generic ensemble concept developed for data clustering which has been proven to be useful for many applications (e.g. bioinformatics).

Selecting algorithms to be based on different objective functions allows the diversity between community structure which would be an added value when the ensemble approach outperform individual algorithms. In addition, the comparison can help to infer the closest performance of individual algorithm to the ensemble one.

The study of multiple relationships and interactions for the same users within one network demonstrated the difference of performance of the MCF in revealing community structure within IBM network. For our test graphs, we observed that the ensemble concept was not usually able to overcome the inherent drawbacks of the nodes and the links structure which was observed when we studied different relationships or interactions.

Also, experimenting different data clustering methods on the hybrid fused communities converts the community detection problem into a data clustering problem which deals with data points. This can benefit predicting links and communities in the future. In addition clustering these points can be mapped to social groups having relations where each group reflects the same level of connectedness and indicates the strength level of the appearance in one community. Generally, we observed that the use of different data clustering methods revealed that the type of the clustering method has a small impact on the performance of the MCF and not a significant one.

We here compared individual algorithms versus the ensemble approach to characterize the structure of communities within the network. We observed that the highest Surprise score and lowest VI score were achieved in static relations represented in the Friendship graph and Managing graph more than active relations represented in the Tagging graph. Compared to individual algorithms, MCF obtained communities with the highest Surprise and lowest VI for Friendship graph. While for the Tagging and Managing graph, individual algorithms perform better although according to the observed network structure when we visualized the networks space, we found that even visually it is hard to define communities for the Tagging and Managing networks.

We observed a strong correlation between the Surprise and VI scores while evaluating the performance of MCF (derived from the combined results of four algorithms) across individual algorithms. Generally, most of the results of the VI measure agrees with the results of the Surprise measure. If the Surprise score can be considered as an optimum strategy for characterizing communities due to the strong correlation with VI, then it should be possible to be used for choosing the best approach across many algorithms that has the highest score. Hence, Surprise is an important parameter of choice for charactering communities in future work.

We suggest that modularity does not act as an appropriate measure for evaluating the communities. The Surprise score here performed more effectively than Modularity. Our results conclude that algorithms that clearly behave poorly across the different networks are the same that produce high modularity values or that use modularity as an objective function to maximize the quality of the communities.

6.10 Summary

In this chapter, we proposed an ensemble approach named as Multi-criteria Community Fusion (MCF) for detecting communities that aims to promote and explore the level of enhancement of the delivered communities using real enterprise network called *IBM Connections* within IBM.

The approach accommodates applications that includes combining various community detection algorithms and multiple network structure. In our example, it is based on the fusion of four community detection algorithms: Walktrap, Infomap, Label propagation and Louvain. Each algorithm is based on different criteria for delivering communities. The experiments include testing the MCF using three clustering method: K-means, Spectral and Hierarchical clustering on different interactions (e.g tagging) and relationships (e.g friendship, managing) between employees to discover hidden structures within the network.

The MCF is evaluated based on a selected evaluation criteria and compared to individual algorithms in order to investigate if the ensemble concept can always behave effectively or not. Results showed that the friendship relation between IBM employees reveals a potentially concrete community structures relative to other tagging and managing relations within the network using the ensemble MCF. The different clustering methods had a slight effect on the performance of MCF. Using Surprise and Variation of Information (VI) as a criteria of evaluating communities obtained no conflict in assessing the results. Hence, they can be considered as a better evaluation criteria than modularity.

We have seen that the fusion of popular existing algorithms does not always lead to more accurate partitions than the ones that delivered by individual algorithm across different networks in the considered dataset. In this way, it is possible to exploit the diversity between partitions and make use of having a possible factor of enhancement for the communities but still an open issue is that not every structure in real-world graphs can cope with the fusion of different stochastic fluctuations.

Chapter 7

Conclusion and Future Directions

In this chapter, we conclude the thesis by highlighting the main contributions of our work. We will also discuss the future work suggested for further improvements.

7.1 Summary of Contributions

The aim of this dissertation is to highlight some challenges of analysing social networks and achieve some progress towards a better understanding of these problems to provide solutions to better deal with them. We tried to achieve this by tackling some of these challenges and by proposing solutions, studies and methodologies to how to deal and overcome the impact of these challenges during mining a network.

Through the thesis, we have addressed some of these challenges. In chapter 4, an approach using machine learning was proposed for predicting the performance or the execution time to analyse a social network. This approach presents a way to deal with evolving and scale-free networks problem. The proposed approach provides an easy way to predict the approximate time taken to analyse a new network (or graph) given the number of nodes and edges within the network. A simulation for the evolving graph is achieved by extracting subgraphs of increasing in size from one network. Then, an analysis is held on some popular graph measures using two different tools like SNAP and Gremlin. Finally, we utilized four different machine learning regression models: MARS, Boosting, M5 and Support Vector Machine. The models were trained and tested over 50 samples of graphs having different sizes in order to select the best model using RMSE an evaluation metric. Our results concluded that MARS outperformed the other three models suggesting that it might be the best suited for addressing this problem. We provided multiple computational models with their coefficients for each graph measure in terms of nodes and edges.

In chapter 5, we have considered the community detection problem and the difficulty of validating the communities derived from real-world social networks. This is due to an unavailable ground truth community structure for most available networks and the existence of different types of community detection algorithms which are used to find the community structure. Thus, an analysis is provided to compare these techniques using partial graphs based on a single relationship extracted from the whole Enron email network (e.g. Netview, To Netview and CC Netview) and to quantify the contribution of attribute similarity by studying the topological properties of these communities using popular measures like size distribution, edge density, transitivity and others. Then an evaluation criteria is used to compare the agreement of different algorithms with each other not to ground truth on the community structure. Different similarity measures are used to recommend that there are specific algorithms and specific relations in the Enron email network which can derive high quality communities. For the Enron network, not all community detection algorithms gave inconclusive results specially on different graph representation. This helps us to conclude that the CC Netview relations followed by To Netview relations provide better community structure based on the topological and evaluation results.

We have also contributed in investigating how to optimize communities without ground truth and the impact of this optimization on passive and active networks in chapter 6 by proposing an approach called Multi-criteria Community Fusion (MCF) based on an ensemble approach. The approach is applied on a real internal enterprise network called IBM Connections. We extracted relations and interactions between employees such as Friendship, Tagging and Managing. We then presented each relation in one graph so that we can derive different graphs with different relations o be able to compare the ensemble community approach based on different graph structures. This approach uniquely combines four community detection algorithms to create an ensemble approach with explicit community structure and adaptable community strength. The MCF enabled us to fuse communities through applying different steps until a hybrid fused communities are derived. Evaluation measures are used to verify the performance of our approach and comparing the communities derived from ensemble approach with the one delivered by each community detection algorithm. Ensemble MCF outperformed other algorithms for the Friendship network within IBM. We have seen that the ensemble approach does not always lead to more accurate results than the ones that delivered by individual algorithm and that it varies across different networks structure. The approach can be used as a generic ensemble approach which can be applied on different networks, different types of algorithms and supports any number of algorithms.

For sure, this work can not cover all topics and the challenges related to social networks analysis. But in the last years, this area has got a lot of research attention. In the following section, we suggest some open problems and questions to be explored for further research.

7.2 Future work

The focus of the work and the main contributions in this thesis are in highlighting the challenges of social network mining and proposing research solutions to tackle these challenges. Our explorations have been focused on different social networks and tried to cover different challenges. The proposed approaches are potentially generic and could be applied on other types of networks. However, our research work in this thesis has the potential to be extended in the future work to include the following:

The proposed analysis and approach based on predictive modeling for measuring the performance in Chapter 4 are based on a single hardware with normal specs. However, we suggest that our analysis and our proposed approach can be applied on multiple hardware with higher specs, such as available CPU resources and RAM size that can achieve higher performance for evolving graphs. It is also interesting to explore the execution time of distributed graph systems that are based on memory approaches like Pregel and GraphLab which are proposed in the literature. Also, since Gremlin showed low performance on a personal machine and given that one of its advantages that it can be integrated with Hadoop (Gremlin/Hadoop) to allow parallel execution of Gremlin scripts as map reduce jobs on a Hadoop infrastructure. We believe that studying the impact on the computational time based on the structure of the network whether it is real-world network or random network can be an interesting path to explore.

Our contribution in Chapter 5 might be a way to identify communities and the community strength without having a "ground truth". The proposed work opens the directions for future research to work on an extra detailed analysis to provide validation of our analysis and our discovered insights through expanding the properties and using a wide range of algorithms with some additional community structure measures. Exploring other email networks and observing if they behave like the Enron network from the community point of view can give a potential insight for the discovery of communities in any email network.

Our proposed approach in chapter 6 for community detection provides an automated ensemble way that aim to improve the communities' quality and comprised the investigation of different network structures. The proposed approach is currently applied on non-overlapping communities for real-world networks, hence, it worth trying it on overlapping communities in real-world networks and discover a deeper level of real communities. One advantage of this approach is that it can can support various ensemble ideas for other types of social networks using multiple number of algorithms and suitable for different community structures. Also, it can be tested on time-stamped networks where communities correspond to different time windows.

7.3 Closing

We believe that the work presented in this thesis contributed to develop new techniques for dealing with some of the challenges and problems in the social network analysis area. The focus is providing new approaches that unlock some of the analytics problems in social networks. We believe that providing solutions for these challenges still needs more research attention and it is as important as paying attention to the area of mining the network itself. Through these contributions, we try to connect and combine different areas through our experiments such as studying most of the popular graph measures used in mining, comparing different analysis tools, applying predictive modeling techniques, exploring various strategies for deriving communities, studying different real world networks and deriving insights about them. We hope that the work in this thesis act as good start for researchers to continue working, investigating the challenges introduced at the beginning and trying to improve or extend the approaches presented in this thesis.

Bibliography

[1] John Adrian Bondy and Uppaluri Siva Ramachandra Murty. *Graph theory with applications*, volume 290. Macmillan London, 1976.

[2] Peter J Carrington, John Scott, and Stanley Wasserman. *Models and methods in social network analysis*, volume 28. Cambridge university press, 2005.

[3] Jure Leskovec, Lars Backstrom, Ravi Kumar, and Andrew Tomkins. Microscopic evolution of social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 462–470. ACM, 2008.

[4] Bryan Klimt and Yiming Yang. The enron corpus: A new dataset for email classification research. In *European Conference on Machine Learning*, pages 217–226. Springer, 2004.

[5] Leonhard Euler. Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, 8:128–140, 1741.

[6] Narsingh Deo. *Graph theory with applications to engineering and computer science.* Courier Dover Publications, 2017.

[7] Duncan J Watts. *Small worlds: the dynamics of networks between order and randomness.* Princeton university press, 1999.

[8] Brett Meador. A survey of computer network topology and analysis examples. *Washington University in St. Louis*, 2008.

[9] Claude Berge. *The theory of graphs.* Courier Corporation, 1962.

[10] Rudolf Fritsch, R Fritsch, G Fritsch, and Gerda Fritsch. *Four-Color Theorem.* Springer, 1998.

[11] BJ Vreugdenhil. *The influence of social network structure on the chance of success of open source software project communities.* PhD thesis, Erasmus Universiteit, 2009.

[12] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[13] David Lazer, Alex Sandy Pentland, Lada Adamic, Sinan Aral, Albert Laszlo Barabasi, Devon Brewer, Nicholas Christakis, Noshir Contractor, James Fowler, Myron Gutmann, et al. Life in the network: the coming age of computational social science. *Science (New York, NY)*, 323(5915):721, 2009.

[14] Sammantha L Magsino. Applications of social network analysis for building community disaster resilience. *Washington, DC: National Academy of Sciences*, 2009.

[15] Wil MP Van der Aalst and Minseok Song. Mining social networks: Uncovering interaction patterns in business processes. *Business Process Management*, 3080:244–260, 2004.

[16] Wayne W Zachary. An information flow model for conflict and fission in small groups. *Journal of anthropological research*, 33(4):452–473, 1977.

[17] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002.

[18] Kurt Mehlhorn and Stefan Naher. Leda: a platform for combinatorial and geometric computing. *Communications of the ACM*, 38(1):96–103, 1995.

[19] D Lusseau, K Schneider, OJ Boisseau, P Haase, E Slooten, and SM Dawson. An undirected social network of frequent associations between 62 dolphins in a community living off doubtful sound. *Behavioral Ecology & Sociobiology*, 54(4):396–405, 2003.

[20] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

[21] MEJ NEWMAN. Network data from mark newmans home page.

[22] S Van Kester. Efficient crawling of community structures in online social networks. 2011.

[23] Michal Jacovi, Ido Guy, Shiri Kremer-Davidson, Sara Porat, and Netta Aizenbud-Reshef. The perception of others: inferring reputation from social media in the enterprise. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, pages 756–766. ACM, 2014.

[24] Matthew Rowe, Miriam Fernandez, Harith Alani, Inbal Ronen, Conor Hayes, and Marcel Karnstedt. Behaviour analysis across different types of enterprise online

communities. In *Proceedings of the 4th Annual ACM Web Science Conference*, pages 255–264. ACM, 2012.

[25] Mark EJ Newman. The structure and function of complex networks. *SIAM review*, 45(2):167–256, 2003.

[26] John W Raymond, Eleanor J Gardiner, and Peter Willett. Rascal: Calculation of graph similarity using maximum common edge subgraphs. *The Computer Journal*, 45(6):631–644, 2002.

[27] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005.

[28] Linton C Freeman, Douglas Roeder, and Robert R Mulholland. Centrality in social networks: Ii. experimental results. *Social networks*, 2(2):119–141, 1979.

[29] Phillip Bonacich. Technique for analyzing overlapping memberships. *Sociological methodology*, 4:176–185, 1972.

[30] Jure Leskovec, Kevin J Lang, and Michael Mahoney. Empirical comparison of algorithms for network community detection. In *Proceedings of the 19th international conference on World wide web*, pages 631–640. ACM, 2010.

[31] Mark EJ Newman. Fast algorithm for detecting community structure in networks. *Physical review E*, 69(6):066133, 2004.

[32] Elizabeth A Leicht and Mark EJ Newman. Community structure in directed networks. *Physical review letters*, 100(11):118703, 2008.

[33] Andrei Broder, Ravi Kumar, Farzin Maghoul, Prabhakar Raghavan, Sridhar Rajagopalan, Raymie Stata, Andrew Tomkins, and Janet Wiener. Graph structure in the web. *Computer networks*, 33(1):309–320, 2000.

[34] Gabor Szabo and Bernardo A Huberman. Predicting the popularity of online content. *Communications of the ACM*, 53(8):80–88, 2010.

[35] Meeyoung Cha, Hamed Haddadi, Fabricio Benevenuto, and P Krishna Gummadi. Measuring user influence in twitter: The million follower fallacy. *Icwsm*, 10(10-17):30, 2010.

[36] Jonathon N Cummings, Brian Butler, and Robert Kraut. The quality of online social relationships. *Communications of the ACM*, 45(7):103–108, 2002.

[37] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, pages 591–600. ACM, 2010.

[38] Ryan Rowe, German Creamer, Shlomo Hershkop, and Salvatore J Stolfo. Automated social hierarchy detection through email network analysis. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, pages 109–117. ACM, 2007.

[39] Jana Diesner and Kathleen M Carley. Exploration of communication networks from the enron email corpus. In *SIAM International Conference on Data Mining: Workshop on Link Analysis, Counterterrorism and Security, Newport Beach, CA*, 2005.

[40] V Varshney and DG Deepak. Analysis of enron email threads and quantification of employee responsiveness. In *Workshop on International Joint Conference on Artificial Intelligence, Hyderabad, India*, 2007.

[41] Jana Diesner, Terrill L Frantz, and Kathleen M Carley. Communication networks from the enron email corpus "it's always about the people. enron is no different". *Computational & Mathematical Organization Theory*, 11(3):201–228, 2005.

[42] Cameron Marlow. Audience, structure and authority in the weblog community. In *International communication association conference*, volume 27, 2004.

[43] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[44] Zoltán Gyöngyi, Hector Garcia-Molina, and Jan Pedersen. Combating web spam with trustrank. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 576–587. VLDB Endowment, 2004.

[45] Jon M Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[46] Josep M Pujol, Ramon Sangüesa, and Jordi Delgado. Extracting reputation in multi agent systems by means of social network topology. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1*, pages 467–474. ACM, 2002.

[47] Christopher S Campbell, Paul P Maglio, Alex Cozzi, and Byron Dom. Expertise identification using email communications. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 528–531. ACM, 2003.

[48] Ulrik Brandes. A faster algorithm for betweenness centrality. *Journal of mathematical sociology*, 25(2):163–177, 2001.

[49] John R Seeley. The net of reciprocal influence. a problem in treating sociometric data. *Canadian Journal of Experimental Psychology*, 3:234, 1949.

[50] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *journal of the Association for Information Science and Technology*, 58(7):1019–1031, 2007.

[51] Jure Leskovec, Lars Backstrom, and Jon Kleinberg. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 497–506. ACM, 2009.

[52] Andrew @articlemccallum2007topic, title=Topic and role discovery in social networks with experiments on enron and academic email, author=McCallum, Andrew and Wang, Xuerui and Corrada-Emmanuel, Andrés, journal=Journal of Artificial Intelligence Research, volume=30, pages=249–272, year=2007 , Xuerui Wang, and Andrés Corrada-Emmanuel. Topic and role discovery in social networks with experiments on enron and academic email. *Journal of Artificial Intelligence Research*, 30:249–272, 2007.

[53] Peter A Gloor and Yan Zhao. Analyzing actors and their discussion topics by semantic social network analysis. In *Information Visualization, 2006. IV 2006. Tenth International Conference on*, pages 130–135. IEEE, 2006.

[54] Manos Tsagkias, Wouter Weerkamp, and Maarten De Rijke. Predicting the volume of comments on online news stories. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1765–1768. ACM, 2009.

[55] David Gibson, Jon Kleinberg, and Prabhakar Raghavan. Inferring web communities from link topology. In *Proceedings of the ninth ACM conference on Hypertext and hypermedia: links, objects, time and space—structure in hypermedia systems: links, objects, time and space—structure in hypermedia systems*, pages 225–234. ACM, 1998.

[56] Mark EJ Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[57] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[58] Jure Leskovec, Kevin J Lang, Anirban Dasgupta, and Michael W Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, 6(1):29–123, 2009.

[59] Santo Fortunato and Claudio Castellano. Community structure in graphs. In *Computational Complexity*, pages 490–512. Springer, 2012.

[60] Information Resources Management Association. *Human Rights and Ethics: Concepts, Methodologies, Tools, and Applications*. Information Science, 2014.

[61] Anupam Biswas and Bhaskar Biswas. A framework for analyzing community detection algorithms. In *Technology Symposium (TechSym), 2016 IEEE Students*, pages 61–66. IEEE, 2016.

[62] William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850, 1971.

[63] Satu Elisa Schaeffer. Graph clustering. *Computer science review*, 1(1):27–64, 2007.

[64] Santo Fortunato. Community detection in graphs. *Physics reports*, 486(3):75–174, 2010.

[65] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 177–187. ACM, 2005.

[66] Jon Kleinberg, Christos Papadimitriou, and Prabhakar Raghavan. Auditing boolean attributes. *Journal of Computer and System Sciences*, 66(1):244–253, 2003.

[67] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-preserving data mining. In *ACM Sigmod Record*, volume 29, pages 439–450. ACM, 2000.

[68] Lars Backstrom, Cynthia Dwork, and Jon Kleinberg. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th international conference on World Wide Web*, pages 181–190. ACM, 2007.

[69] Jiwon Seo, Sini Guo, and Monica S Lam. Socialite: Datalog extensions for efficient social network analysis. In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, pages 278–289. IEEE, 2013.

[70] Yonathan Perez, Rok Sosič, Arijit Banerjee, Rohan Puttagunta, Martin Raison, Pararth Shah, and Jure Leskovec. Ringo: Interactive graph analytics on big-memory

machines. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1105–1110. ACM, 2015.

[71] Sameh Elnikety and Yuxiong He. System support for managing large graphs in the cloud. In *Proceedings of the NSF Workshop on Social Networks and Mobility in the Cloud*. Citeseer, 2012.

[72] Cong Yu. Beyond simple parallelism: Challenges for scalable complex analysis over social data. In *Proceedings of the NSF Workshop on Social Networks and Mobility in the Cloud*, 2012.

[73] Grzegorz Malewicz, Matthew H Austern, Aart JC Bik, James C Dehnert, Ilan Horn, Naty Leiser, and Grzegorz Czajkowski. Pregel: a system for large-scale graph processing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 135–146. ACM, 2010.

[74] Yucheng Low, Danny Bickson, Joseph Gonzalez, Carlos Guestrin, Aapo Kyrola, and Joseph M Hellerstein. Distributed graphlab: a framework for machine learning and data mining in the cloud. *Proceedings of the VLDB Endowment*, 5(8):716–727, 2012.

[75] Aapo Kyrola, Guy Blelloch, and Carlos Guestrin. Graphchi: Large-scale graph computation on just a pc. In *Presented as part of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12)*, pages 31–46, 2012.

[76] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. Spark: Cluster computing with working sets. *HotCloud*, 10:10–10, 2010.

[77] Hadoop: An open-source framework for distributed processing of large datasets. `http://hadoop.apache.org/`.

[78] Joseph E Gonzalez, Yucheng Low, Haijie Gu, Danny Bickson, and Carlos Guestrin. Powergraph: Distributed graph-parallel computation on natural graphs. In *Presented as part of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12)*, pages 17–30, 2012.

[79] Jiwon Seo, Jongsoo Park, Jaeho Shin, and Monica S Lam. Distributed socialite: a datalog-based language for large-scale graph analysis. *Proceedings of the VLDB Endowment*, 6(14):1906–1917, 2013.

[80] D UUman Jeffrey. Principles of database and knowledge-base systems, 1989.

[81] Giraph. `http://giraph.apache.org/`.

[82] Tyson Condie, David Chu, Joseph M Hellerstein, and Petros Maniatis. Evita raced: metacompilation for declarative networks. *Proceedings of the VLDB Endowment*, 1(1):1153–1165, 2008.

[83] Iris: An open-source datalog engine. `http://www.iris-reasoner.org/`.

[84] Logicblox. `http://www.logicboxsoftware.com/`.

[85] Marko A Rodriguez. The gremlin graph traversal machine and language (invited talk). In *Proceedings of the 15th Symposium on Database Programming Languages*, pages 1–10. ACM, 2015.

[86] Jure Leskovec and Rok Sosič. Snap: A general-purpose network analysis and graph-mining library. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(1):1, 2016.

[87] Jure Leskovec and Rok Sosič. Snap.py: SNAP for Python, a general purpose network analysis and graph mining tool in Python. `http://snap.stanford.edu/snappy`, June 2014.

[88] Max Kuhn and Kjell Johnson. *Applied predictive modeling*. Springer, 2013.

[89] Frank Harrell. *Regression modeling strategies: with applications to linear models, logistic and ordinal regression, and survival analysis*. Springer, 2015.

[90] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[91] R Core Team et al. R: A language and environment for statistical computing. 2013.

[92] Max Kuhn et al. Caret package. *Journal of statistical software*, 28(5):1–26, 2008.

[93] Duncan J Watts and Steven H Strogatz. Collective dynamics of small-worldnetworks. *nature*, 393(6684):440, 1998.

[94] W Fan and KH Yeung. Similarity between community structures of different online social networks and its impact on underlying community detection. *Communications in Nonlinear Science and Numerical Simulation*, 20(3):1015–1025, 2015.

[95] Günce Keziban Orman, Vincent Labatut, and Hocine Cherifi. Comparative evaluation of community detection algorithms: a topological approach. *Journal of Statistical Mechanics: Theory and Experiment*, 2012(08):P08001, 2012.

[96] Leto Peel. Estimating network parameters for selecting community detection algorithms. In *Information Fusion (FUSION), 2010 13th Conference on*, pages 1–8. IEEE, 2010.

[97] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[98] Jörg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74(1):016110, 2006.

[99] Mark EJ Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical review E*, 74(3):036104, 2006.

[100] Luca Donetti and Miguel A Muñoz. Improved spectral algorithm for the detection of network communities. *arXiv preprint physics/0504059*, 2005.

[101] Stijn Van Dongen. Graph clustering via a discrete uncoupling process. *SIAM Journal on Matrix Analysis and Applications*, 30(1):121–141, 2008.

[102] Pascal Pons and Matthieu Latapy. Computing communities in large networks using random walks. In *International Symposium on Computer and Information Sciences*, pages 284–293. Springer, 2005.

[103] Martin Rosvall and Carl T Bergstrom. An information-theoretic framework for resolving community structure in complex networks. *Proceedings of the National Academy of Sciences*, 104(18):7327–7331, 2007.

[104] Martin Rosvall and Carl T Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123, 2008.

[105] Guojun Gan, Chaoqun Ma, and Jianhong Wu. *Data clustering: theory, algorithms, and applications*. SIAM, 2007.

[106] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[107] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical review E*, 76(3):036106, 2007.

[108] Mark EJ Newman. Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 103(23):8577–8582, 2006.

[109] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical review E*, 78(4):046110, 2008.

[110] Jure Leskovec, Kevin J Lang, Anirban Dasgupta, and Michael W Mahoney. Statistical properties of community structure in large social and information networks. In *Proceedings of the 17th international conference on World Wide Web*, pages 695–704. ACM, 2008.

[111] Roger Guimera, Leon Danon, Albert Diaz-Guilera, Francesc Giralt, and Alex Arenas. Self-similar community structure in a network of human interactions. *Physical review E*, 68(6):065103, 2003.

[112] Andrea Lancichinetti, Mikko Kivelä, Jari Saramäki, and Santo Fortunato. Characterizing the community structure of complex networks. *PloS one*, 5(8):e11976, 2010.

[113] Hélio Almeida, Dorgival Guedes, Wagner Meira Jr, and Mohammed J Zaki. Is there a best quality metric for graph clusters? In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 44–59. Springer, 2011.

[114] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985.

[115] LNF Ana and Anil K Jain. Robust data clustering. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–II. IEEE, 2003.

[116] Leon Danon, Albert Diaz-Guilera, Jordi Duch, and Alex Arenas. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(09):P09008, 2005.

[117] Jitesh Shetty and Jafar Adibi. The enron email dataset database schema and brief statistical report. *Information sciences institute technical report, University of Southern California*, 4, 2004.

[118] Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research. *InterJournal, Complex Systems*, 1695(5):1–9, 2006.

[119] Rong Qian, Wei Zhang, and Bingru Yang. Detect community structure from the enron email corpus based on link mining. In *Intelligent Systems Design and Applications, 2006. ISDA'06. Sixth International Conference on*, volume 2, pages 850–855. IEEE, 2006.

[120] Farnaz Moradi, Tomas Olovsson, and Philippas Tsigas. An evaluation of community detection algorithms on large-scale email traffic. *SEA*, 7276:283–294, 2012.

[121] Waqas Nawaz, Kifayat-Ullah Khan, and Young-Koo Lee. A multi-user perspective for personalized email communities. *Expert Systems with Applications*, 54:265–283, 2016.

[122] Alexander Strehl and Joydeep Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3(Dec):583–617, 2002.

[123] Aik Choon Tan and David Gilbert. Ensemble machine learning on gene expression data for cancer classification. 2003.

[124] Anuska Ferligoj and Vladimir Batagelj. Direct multicriteria clustering algorithms. *Journal of Classification*, 9(1):43–61, 1992.

[125] Mursel Tasgin, Amac Herdagdelen, and Haluk Bingol. Community detection in complex networks using genetic algorithms. *arXiv preprint arXiv:0711.0491*, 2007.

[126] Peter J Mucha, Thomas Richardson, Kevin Macon, Mason A Porter, and Jukka-Pekka Onnela. Community structure in time-dependent, multiscale, and multiplex networks. *science*, 328(5980):876–878, 2010.

[127] Gergely Palla, Albert-László Barabási, and Tamás Vicsek. Community dynamics in social networks. *Fluctuation and Noise Letters*, 7(03):L273–L287, 2007.

[128] Petter Holme and Jari Saramäki. Temporal networks. *Physics reports*, 519(3):97–125, 2012.

[129] Leonard Kaufman and Peter J Rousseeuw. *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons, 2009.

[130] Sung-Hyuk Cha. Comprehensive survey on distance/similarity measures between probability density functions. *City*, 1(2):1, 2007.

[131] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

[132] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

[133] Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, pages 849–856, 2002.

[134] Stephen C Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, 1967.

[135] Rodrigo Aldecoa and Ignacio Marín. Surprise maximization reveals the community structure of complex networks. *Scientific reports*, 3, 2013.

[136] Rodrigo Aldecoa and Ignacio Marín. Deciphering network community structure by surprise. *PloS one*, 6(9):e24195, 2011.

[137] Marina Meilă. Comparing clusteringsan information based distance. *Journal of multivariate analysis*, 98(5):873–895, 2007.

[138] P Jaccard. Jura, bulletin societevandoise des sciences naturelles. *Étude comparative de la distribuition florale dans une portion des Alpes et des*, 37:547–579, 1901.