Summer 5-2014

COMMUNITY DETECTION AND INFLUENCE MAXIMIZATION IN ONLINE SOCIAL NETWORKS

Kanna Gharib Al-Falahi

United Arab Emirates University

College of Information Technology

COMMUNITY DETECTION AND INFLUENCE MAXIMIZATION IN ONLINE SOCIAL NETWORKS

Kanna Gharib Al-Falahi

This dissertation is submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Under the direction of Dr. Yacine Atif

May 2014

DECLARATION OF ORIGINAL WORK

I, Kanna Gharib Al-Falahi, the undersigned, a graduate student at the United Arab Emirates University (UAEU) and the author of the dissertation entitled “Community Detection and Influence Maximization in Online Social Networks”, hereby solemnly declare that this dissertation is an original research work done and prepared by me under the guidance of Dr. Yacine Atif, in the College of Information Technology at UAEU. This work has not been previously formed as the basis for the award of any academic degree, diploma or similar title at this or any other university. The materials borrowed from other sources and included in my dissertation have been properly cited and acknowledged.

Student’s Signature: .... Date: ....

ABSTRACT

The detecting and clustering of data and users into communities on the social web are important and complex issues in order to develop smart marketing models in changing and evolving social ecosystems. These marketing models are created by individual decision to purchase a product and are influenced by friends and acquaintances. This leads to novel marketing models, which view users as members of online social network communities, rather than the traditional view of marketing to individuals.

This thesis starts by examining models that detect communities in online social networks. Then an enhanced approach to detect community which clusters similar nodes together is suggested. Social relationships play an important role in determining user behavior. For example, a user might purchase a product that his/her friend recently bought. Such a phenomenon is called social influence and is used to study how far the action of one user can affect the behaviors of others. Then an original metric used to compute the influential power of social network users based on logs of common actions in order to infer a probabilistic influence propagation model. Finally, a combined community detection algorithm and suggested influence propagation approach reveals a new influence maximization model by identifying and using the most influential users within their communities. In doing so, we employed a fuzzy logic based technique to determine the key users who drive this influence in their communities and diffuse a certain behavior. This original approach contrasts with previous influence propagation models, which did not use similarity opportunities among members of communities to maximize influence propagation. The performance results show that the model activates a higher number of overall nodes in contemporary social networks, starting from a smaller set of key users, as compared to existing landmark approaches which influence fewer nodes, yet employ a larger set of key users.

Keywords: Social networks, community detection, social influence, influence maximization, influence metrics, fuzzy logic.

ملخص

يعد اكتشاف و تجميع المستخدمين و البيانات في الويب الإجتماعي من القضايا المهمة و المعقدة التي تهدف لتطوير نماذج تسويق ذكية ضمن النظام الإجتماعي المتغير و المتطور. تعتمد نماذج التسويق هذه على ملاحظة قرارات الأفراد لشراء منتج معين أو تبني فكرة معينه و التي تتأثر بالتوصيات من مجتمعات أصدقاء و معارف هؤلاء الأفراد. هذه الملاحظة تقود لنماذج تسويقية مبتكرة و التي بدورها تعرض المستخدمين كأعضــــاء في مجتمعات شبكة إجتماعية على الإنترنت بدلاً من النظرة التقليدية للتسويق الإنفرادي للأفراد.

في هذه الأطروحة، سنبدأ بكشف و معاينة النماذج المستخدمة لاكتشاف المجتمعات في الشبكات الإجتماعية على الإنترنت. و من ثم سنقترح مساهمتنا الأولى و هي نهج يهدف لتعزيز إكتشاف المجتمعات من خلال تقنية تهدف لتجميع الأشخاص المتشابهين معاً. في داخل المجتمعات تلعب العلاقات الإجتماعية دورها هاماً في تحديد سلوكيات الأفراد. على سبيل المثال، قد يقدم فرد من الأفراد على شراء منتج قد سبق و أن اشتراه صديقه مؤخراً. تسمى هذه الظاهرة بالتأثير الإجتماعي، و الذي يستخدم لدراسة مدى تأثير سلوك فرد من أفراد المجتمع على سلوك جيرانه و بقية أفراد المجتمع في الشبكة الإجتماعية. مساهمتنا الثانية عبارة عن طريقة غير مسبوقة لحساب قوة تأثير أفراد الشبكة الإجتماعية على غيرهم بناءاً على سجلات لسلوكيات الأفراد المتشابهة و التي من خلالها نستنتج احتمالية انتشار نفوذ فرد معين. مساهمتنا الأخيرة تجمع الخوارزمية المقترحة لتعزيز الكشف عن المجتمعات و نهج نشر التأثير الإجتماعي لنكشف عن نموذج جديد لزيادة التأثير الإجتماعي من خلال تحديد الأفراد الأكثر تأثيراً في مجتمعاتهم. للقيام بذلك نقوم باستخدام تقنية المنطق الضبابي لتحديد الأفراد الرئيسيين و الذين يقودون النفوذ داخل مجتمعاتهم لنشر سلوك أو إبتكار معين. هذا النهج الغير مسبوق يختلف عن نماذج انتشار النفوذ التقليدية و التي لا تغتنم فرص التشابه الموجودة بين أفراد المجتمعات و التي تؤدي بالتالي إلي زيادة إنتشار تأثير معين. تظهر نتائج الأداء أن نموذجنا "يحفز" عدداً أكبر من الأفراد على تبني فكرة معينة أو سلوك معين في الشبكات الإجتماعية المعاصرة بدءاً من مجموعة صغيرة من الأفراد الرئيسيين مقارنة بالنماذج التقليدية و التــــي تحفز عدداً أقل من الأفــراد كــما أنها تستخدم عــــدداً أكبر من الأفـراد كـنقطة انطــــلاقة لانتشار التأثير الإجتماعي.

كلمــات مفتاحـــية: شبكــات إجتماعية، إكتشاف المجتمعــات، التأثير الإجتماعي، زيـــادة التأثير الإجتماعي، قياس التأثير، المنطق الضبابي.

ACKNOWLEDGMENTS

I would like to thank my supervisor, Dr. Yacine Atif, for always inspiring me to pursue excellence and for assisting me throughout my studies and research. His encouragement, guidance, and positive attitude were pivotal in my study. I am deeply indebted to him.

Special thanks to the thesis committee for their guidance, support and assistance throughout preparation of this thesis/dissertation. Dr. Mohammed Shehab, Dr. Saad Harus and Dr. Nadir Mohammed, thank you so much.

Finally, I would like to thank my mother, brothers and sisters for their continuous support. I am grateful for their love and for being by my side when completing this thesis. It would not have been possible without their support.

DEDICATION

To my mother, thank you and I love you

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

During the past decade, social networks have increased dramatically along with rapid developments in Web 2.0 applications. Millions of people participate in social networks such as Facebook, Twitter and MySpace. Facebook in particular, accumulated more than 1 billion active user in 2012[1]. As the number of users increases, the complexity induced when evaluating social networks increases too. Moreover, the bewildering number of options, which are continuously expanding the scope of these networks across social, business[2] and even governmental[3] spaces, sparked the need for criteria or measures to understand the current and future prospects for social networks [4]. However these networks tend to be large and complex. Therefore to understand and summarize whole network properties, there is an evolving need to harness this complexity by organizing the network into communities. This rising online organization has revealed the power of social influence in today's viral marketing campaigns. In recent years, the study of influence propagation has gained a lot of attention in both academic and business contexts to maximize the number of users who follow a particular action, behavior or product.

[1]One Billion People on Facebook, Facebook NewsRoom, https://newsroom.fb.com/news/2012/10/one-billion-people-on-facebook/ (accessed April 20, 2014)

[2]Dell Outlet, Twitter, http://twitter.com/delloutlet (accessed April 20, 2014)

[3]Barack Obama, Facebook, http://www.facebook.com/barackobama (accessed April 20, 2014)

1.1 Motivation

Network science, which arguably had its beginnings in the 1700s with Euler's 'Seven Bridges of Knigsberg' [5], has passed through a number of stages, including the appearance of graph theory [6], the sociogram (i.e. graphical representation of social links), and the advent of social network analysis [7], culminating in the recent boom and solidification as a discipline. Just after, some of the most important recent innovations, such as the development of scale-free networks [8], the first social networking sites started to appear [9], and within less than a decade, Facebook had 51% of the world's online population as active users [10].

Social networks are essentially built by groups of people who share similar interests, backgrounds and activities. In social networks, people can communicate with each other in many ways. They can share and upload profiles with images, videos and texts. Social networks consist of nodes that are considered as *the actors* of the network. These nodes might represent a user, a company, etc. The nodes are interlinked through *connections*, which represent the relationships between them as friendship, partnership, etc. The cardinality of nodes expands dynamically, especially on the web, as new nodes and profiles are created continuously, while further populating the social web [11]. Social web is based on a set of relations that connect people through the World Wide Web. *Hubs* are one social network phenomena, defined simply by nodes that have a large number of links [12]. An example for a hub would be a Facebook page with many incoming links. Figure 1.1 shows a graphic view of a typical social network involving profiles and web objects with a typical hub node. Such hub nodes are influential in spreading trends across social networks, due to the high number of connections they have.

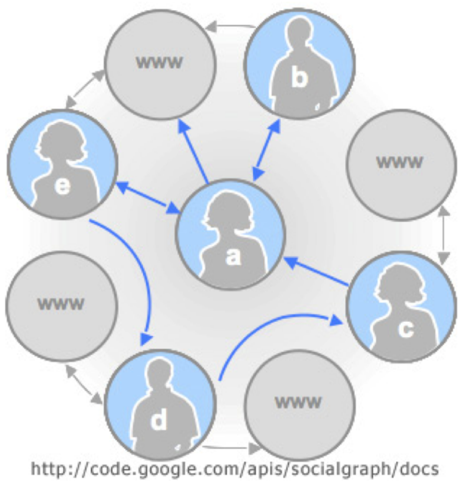


Figure 1.1: Social Graph

Social networks have gained significant importance due to their great impact on people's everyday lives. We increasingly rely on recommendations and influence from friends and acquaintances to choose the best products to buy. Nowadays, people generally depend on the Internet to make decisions. However, the Internet alone cannot provide users with sufficient support for their decision-making processes as it contains a wide number of products and services to choose from. The current online social structure is paving the way for social networks to take a pivotal role in generating recommendations, based on social influence, and recommender systems integration [13]. These trends are expected to promote new intuitions and observations that would not have been achieved through traditional recommendations. Moreover, the need for an effective approach to maximize the number of individuals who will adopt a recommended product is soaring as businesses compete to find innovative viral marketing strategies to advertise their products with minimal effort and small budgets. When it comes to merchants, the immediate and tangible economic benefits of a successful recommendation is expressed in terms of increasing sales and creating revenue. On the other hand, when it comes to users (i.e. potential buyers), nowadays, they are often overwhelmed with a multitude of choices and options in their online business experiences, while at the same time they have limited resources and free time to invest in the selection process. Finding an accurate and cost-effective solution to address these challenges will increase the precision and efficiency of viral marketing over traditional means of propagating marketing campaigns for products and services.

We can illustrate the adoption process for products in a social context through the following scenario. Assume Ahmed wants to go to the cinema with his friend Saeed. But they do not know which movie to watch, knowing only that Saeed does not like action movies. They start to ask their friends online using Facebook or Twitter. A very close friend of Ahmed (trusted by Ahmed), Mohammed, sends his recommendation to watch 'Les Misérables' as he just saw it in the theater last week and he thinks the movie has a really good story to tell. Mohammed was probably influenced by other friends about this movie. Saeed's friends also may recommend movies too. Based on the most suitable level of influence from friends and their positive opinions, that also fit with Ahmed and Saeed's preferences they determine to watch 'Les Misérables'.

In the example above we can see that there are different aspects that encourage people to accept (i.e. adopt) a recommended product. Such as the strength of a relationship or the level of trust between people in the social network. This affinity between people reveals the power of community structures in matching individual behaviors to actions.

In this thesis, we investigate community detection solutions that support social influence. In doing so, we look into online social networks to find sets of influential and important users (seed sets), who are interested in a product, to the point that they will provide a positive review and recommend it to their friends. Then, their friends will embrace the product (given the influential power of the recommenders) and will also recommend it to their friends, and so on. We want to build an influential system that will increase the possibility of recommending and adopting products initiated form a small set of important (or influential) users in contemporary online social networks. Our proposal aims also to solve the famous 'cold start' problem, faced by companies when starting a marketing campaign. A set of active and influential users can propagate the adoption of a new product. This seed set of active users will maximize word-of-mouth recommendations as they have high connections and influential power.

Traditional recommender systems [14, 15, 16, 17, 18, 19, 20, 21, 22] do not take the social relationships between users into consideration even though studies measuring the importance of social influence have been conducted [23, 24]. One chapter [25] provides a detailed study of traditional and current recommender systems as well as the challenges they face. When friends tend to recommend products, other friends will accept these recommendations most of the time, as they trust each other. Businesses which adopted social influence in their marketing campaigns were successful. For example, Hotmail used social influence to achieve 12 million subscribers in just 18 months with a marketing budget of only $50,000. Hotmail spread all over the World even in countries where they did not advertise such as Sweden and India. This shows that relationships are powerful when making decisions on buying or adopting products or ideas. Another real example of influence can be seen in the effects of obesity. Christakis and Fowler [26] provided a study of 12,067 people from 1971 to 2003. The results of their study shows that having an obese friend will increase an individual's chance of obesity by 57%. While having an obese sibling or spouse increases the chance of obesity by 40% and 37% respectively. Other real examples of this propagation can be seen in the spread of viruses (like the Melissa computer worm), diseases (like mad-cow disease), fashions (like floral prints trending in Spring 2013) and the adoption of new technologies.

Social Influence has been studied in different disciplines and has it's historical roots in sociology, through studying opinion formation, diffusion of innovations [27, 28] and economics, where social influence shows how individuals are inclined to coordinate decisions [29, 30]. Recently, digital social influence research has emerged and started to attract more attention due to the availability of many important applications. For example, computer scientists developed models of social influence to support applications such as viral marketing [31, 3, 32], the spread of online news [33, 34], and the growth of online communities [35].

Most current applications for online social networks focused on increasing the accuracy of item rating. In this thesis, we investigate the increase of word-of-mouth recommendations through an influence propagation model which applies novel influence maximization techniques.

1.2 Terms, Concepts And Properties of Social Networks

In this section, we discuss different concepts and properties that are related to social networks and reveal some related work done on analytical issues. The main goal of social networks is to connect people so that each user in a social network can establish a link with other users in that network. These connections and relations in social networks are modeled as a graph G which is defined as a pair of sets $G = (V,E)$, where V is a set of N nodes $V = \{v_1, v_2, ..., v_n\}$ and $E \subseteq V \times V$ is a set of edges that connect pairs of nodes v_i, v_j within V [36]. In other words $V \times V$ is an adjacency matrix $E = [E_{ij}]\, i, j \in V$, where $V_{ij} \in \{0,1\}$ represents the availability of an edge from node i to node j. The edge weight $E_{ij} > 0$ represents the intensity of interaction and the graph $G(V,E)$ in that case is called a weighted graph. The graph is directed if $E_{ij} \neq E_{ji}$ and undirected if $E_{ij} = E_{ji}$ for all $i, j \in V$ [37].

1.2.1 Basic Concepts of Online Social Networks

User Profile

Most social networks provide their functionalities for free to the users. Though some social networks need users to register in order to gain access to full facilities. Personal information about each user is stored in his/her profile, where a profile is a collection of user information that shapes the user's identity and other personal attributes such as interests [38].

User Connections

The main goal of social networks it to connect people, thus each user in a social network can establish a link with other users in the network. Figure 1.2 shows the types of relationships that occur in social networks. An example would be the concept of ‘follow-me’, in Twitter where a user (creator) can follow other users (targets). A full connection between the creator and the target is established if both are following each other. In the case of the Twitter example, a full connection will allow additional functionalities such as the ability to send private messages between users. Users establish these connections in order to follow each other’s contributions, especially if they share similar interests.

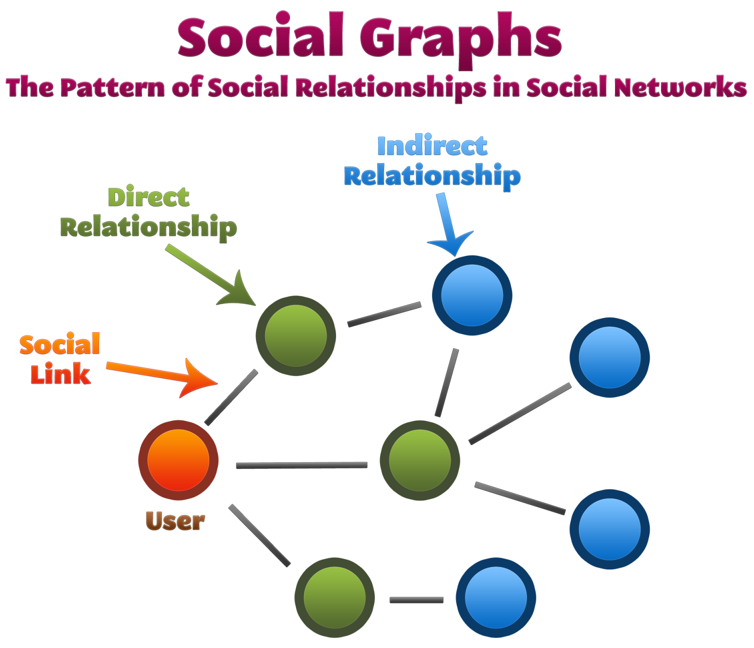


Figure 1.2: Social Graph: Patterns of Social Relationships Between People

Profile Privacy

Many social networks allow any user to view other users’ profiles, though some social networks such as Facebook provide users with privacy levels that allow them to access only a particular group of profiles.

1.2.2 Properties of Online Social Networks

Connectivity

That is the degree to which the nodes of a network are directly connected [36]. When a network has high connectivity this means it has a high ratio of edges to the number of nodes[36]. The connectivity is calculated using the following equation [36]:

$$C = \frac{|E|}{|V|(|V|-1)} \tag{1.1}$$

Where E is the number of edges and V is the number of nodes in the network

Network Diameter

The diameter of a network is the length of the longest path between two nodes. In social networks, the diameter is small and averaged by the number six as in small world phenomenon [39]. This will affect the processes that take place in the network. For example there can be a fast spread of information such as ‘rumors’ [40].

Large-Scale

Each network has basic properties such as: network order, represented by the number of nodes in the network; the size, that represents the number of edges in the network; and the node degree, which represents the number of edges that are connected to a node. OSNs are large-scale networks with high order and size that may reach millions. For example in Twitter, the nodes of celebrities such as Katy Perry, Justin Bieber, Barack Obama, Lady Gaga, and Taylor Swift have more than 40 million followers. LinkedIn has more than 90 million nodes, having a new user joining every second [41].

Network Clustering

The idea of clusters or cliques is very common in social networks. Clusters are groups of friends who know each other. The degree to which nodes are able to cluster together can be measured by the clustering coefficient. In general the clustering coefficient C is based on the number of closed triples in a network (a set of three nodes connected to each other's 'triangles') and it can be calculated as in Equation [42]:

$$C = \frac{3 \times number\,of\,triangles}{number\,of\,connected\,triples\,of\,vertices} \tag{1.2}$$

For example the clustering coefficient C for the network below can be measured as follows in Figure 1.3:

$$C = \frac{3 \text{ x } 1}{5} = \frac{3}{5}$$

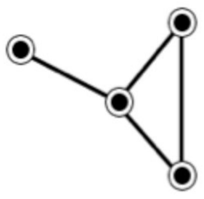


Figure 1.3: Example On Clustering Coefficient

Power Law Degree Distribution

The degree of a node represents the number of edges connected to that node [36]. A distribution function $P(K)$ gives the probability that a node selected at random has degree K [36]. Plotting the $P(K)$ function for a network, generates a histogram of degree distribution of nodes similar to the one shown in Figure 1.4. Note that the distribution has a long right tail as shown in Figure 1.4. The long right tail indicates that in social networks, most nodes have a low degree, whereas a small number of nodes known as '*hubs*' have a high degree. This is fairly true for social networks. Many studies [4, 43, 44, 45] showed that OSNs follows the power law degree distribution.

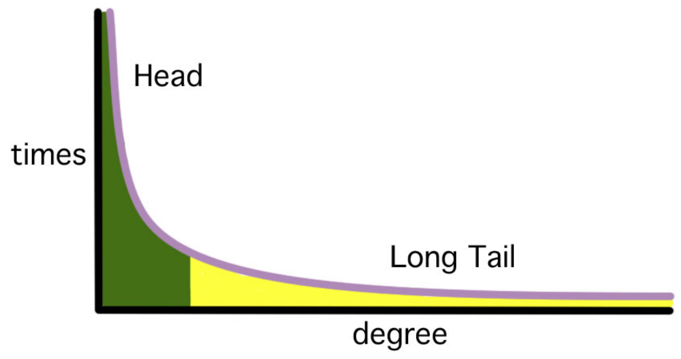


Figure 1.4: Histogram of Degree Distribution of Nodes

Reputation and Trust

In social networks, trust relationship between acquaintances and friends is paramount. Amazon for example, has a recommender system that we trust. Another example would be eBay as it uses the seller rating system in order to allow sellers and buyers to build reputation. Digg is an example for rating web content, where people 'digg' articles they like from all over the web and the most popular articles are promoted on the front page of the website so million of people can view them. Much research has been done in the area of social network trust, but in order to build a robust reputation and trust, a deeper understanding of social topology is required [4]. Understanding social network topology will help in identifying properties about the different members of the network. For example, the location of the member can be used to infer the power and reputation of that member. These members could be identified easily through the number of connections they have with other nodes in the network. This information could support an automatic reputation system in the future, instead of using a manual rating system provided by the users.

Users' Interests

Users in social networks tend to navigate their neighbors' profiles because they may find things of interest there. Systems like Delicious (social bookmarking) allow users to bookmark web links of interests and share them with their friends or explore the bookmarks of other users. StumbleUpon also helps users to discover and rate web pages, photos and videos based on their own interests. This content is recommended by the users' friends or other people who share the same interest [46]. It gives them the choice to 'like' or 'not-like' the content recommended, which also increases the quality of content recommendations.

Some of the above properties are common in complex networks, for example large-scale and power law distribution. These OSNs properties are used to analyze issues pertaining to social influence. Another important property of OSNs that is of benefit in social influence analysis is the ability to retrieve OSN data easily through APIs, while in real world social networks, a substantial physical effort is needed in order to collect this data. The opportunity for data extraction and analysis in OSNs has encouraged an increased research importance to model OSNs.

Thus, understanding the structure of social networks helps in evaluating the strengths, weaknesses, opportunities and threats associated with them. Much work have been done in the field of social networks analytics. One of the most popular papers is by Milgram, 'The Small-World Problem' [39], where the earliest experiments about 'six degrees of separation' were investigated. Milgram studied the average path length for social networks in the United States and suggested that we live in a small world. Watts also studied the mathematical analyses of the small world structure [47] as he examined the small world systems and discussed the problems of measuring the distances in the social world and studied examples of real small-world networks.

1.3 Research Issues and Contributions

The problems addressed in this research are twofold: maximizing social word-of-mouth recommendations by spreading influence and increasing the rate of adopting recommendations in online social networks. This thesis addresses the following research issues:

1. Detecting communities in online social networking

2. Finding key nodes within each community (resulting from Step 1)

3. Constructing the seed set (made up of top users generated from Step 2) which is used to spread influence over an online social network.

Figure 1.5 shows a framework for our proposed model.

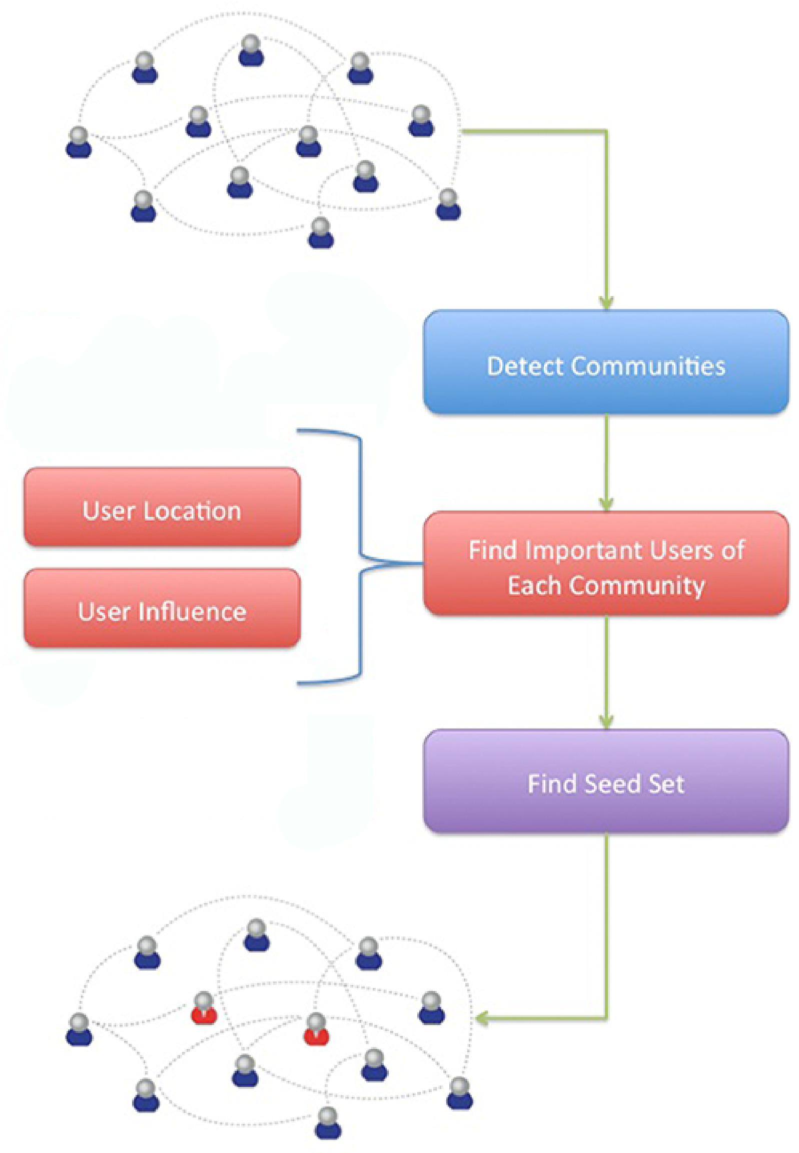


Figure 1.5: Community Aware Influence Maximization

1.3.1 Community Detection

Contribution 1: Survey and Evaluation of Existing Community Detection Approaches

The ability to find groups of interest in a network can help in many ways to provide different services such as targeted advertisements. The problem researchers face here is how to find groups of users who share similar interests or have a high level of connections between them. A critical review of existing approaches proposed in the field of community detection was reviewed and their strengths and limitations were addressed. There was also an analytical study and a performance evaluation of these approaches to contrast and compare different aspects of the techniques employed. In Chapter 2 we provide a thorough survey of the results.

Contribution 2: Similarity-CNM Based Community Detection

One of our research goals is to provide users with an online means to identify or build communities on the web. The focus is on the social web and providing new techniques to detect and build robust communities. We extended the CNM algorithm to use the Jaccard Similarity Measure to first infer an isomorphic graph from the original network, resulting in what we label as a similarity social network or a virtual social network. Our technique showed that by pre-processing the original network, we can derive better quality community structures. Chapter 2 shows more details about this algorithm.

Contribution 3: Jaccard Similarity Based Community Detection

Another algorithm to build strong communities over the web is the ECD-Jaccard Algorithm (ECD refers to "Enhancing Community Detection") which enriches the virtual social network with weights on edges and then applies a quality-optimized version of the CNM algorithm [48] to detect communities. Our technique showed that by pre-processing the original network, we can derive better quality community structures. More details about this algorithm are given in Chapter 2.

1.3.2 Social Influence

Contribution 4: Classification and Evaluation of Existing Social Influence Approaches

Social influence was addressed by first evaluating different models used to measure influence probability. Research carried out in this area is sparse. Besides our efforts to summarize the state of the art surrounding social influence in online social networks, we also surveyed and evaluated different approaches and have clustered them into an original categorization framework to understand commonalities and distinguish differences. This is discussed in Chapter 3.

Contribution 5: Common Actions Based Estimation of Influence Weights

The historical node activity is one of the most valuable estimates of the influence of a node on other nodes in the network. The Jaccard Coefficient Based on Common Actions is proposed to estimate such influence weights in social networks. Further details are described in Chapter 3.

1.3.3 Community Aware Influence Propagation

Contribution 6: Synthetic Communities for Influence Propagation

In this thesis, the influence maximization issue is addressed where we find a set of k nodes which are the most efficient at spreading influence in the network. The community-based influence algorithm starts by detecting communities in the social network as a pre-processing step, in order to group similar nodes together. This step is the basis for influence propagation, as discussed in Chapter 4.

Contribution 7: Fuzzy Decision-Making Approach to Find Key Influential Nodes

To decide which nodes are the most influential, we use a Fuzzy logic inspired method, which computes and selects influential nodes based on both their central location and influence weight. As far as we know, using Fuzzy logic to find the most influential nodes in online social networks has not been explored before. Chapter 4 explains the methodology.

1.4 Thesis Organization

The rest of this thesis is organized as follows. In Chapter 2, we introduce basic concepts related to clustering in social webs, enabling algorithms and present state-of-the-art research in this field. Different basic algorithms are discussed and are used to group social networks nodes into clusters that share similarities. Examples of such algorithms are Linked-based, K-means, Robust Clustering Using Links (ROCK) and Density-Based Spatial Clustering of Applications with Noise (DB-SCAN). We show how each algorithm works and discuss potential advantages and shortcomings. These algorithms are then compared against each other and their ability to accurately identify communities of interest based on social web data is discussed. This is illustrated and discussed in our findings, which involved clustering in online social networks. Finally, two proposed approaches are presented, a Similarity-CNM algorithm and an ECD-Jaccard algorithm, to enhance community detection processes for both unweighted and weighted social networks. Our experimental evaluation study reveals interesting tradeoffs and the effectiveness of the proposed approaches.

In Chapter 3, we introduce social influence and discuss the metrics used to measure influence probability. Different considerations in the field of modeling influences are provided in this chapter. Means are revealed to maximize social influence by identifying and using the most influential users in a social network. We also surveyed existing social influence models and clustered them into an original categorization framework and applied experiments to compare the Linear Threshold (LT) and Independent Cascade (IC) Models.

In Chapter 4, our community aware influence maximization proposal, based on combining both community detection and social influence is presented. In particular, we discuss how we can maximize the influence between social network nodes using enhanced community detection techniques. To explain our proposal we first propose our method of estimating influence weights based on historical common actions in online social networks. Then we discuss our novel contribution by using fuzzy logic inspired method to find the most influential nodes in a social network. Finally, our experimental results for applying our proposed algorithm in real-world social network are discussed. Our findings show the effectiveness of the proposal.

Chapter 5, provides concluding remarks and discusses future directions for further research.

Chapter 2

Enhancing Community Detection in Social Networks

2.1 Introduction

Since its inception, the web has evolved into a huge repository of information of all kinds. The current semantic enrichment of web data and the automation of web services [49], have given rise to novel models for retrieving and analyzing information, particularly in social contexts. In these domains, web content is increasingly collaboratively generated and communicated across blogs, feeds and other social forums. The opportunity to analyze similarities within these social contexts empowers web experiences to recommend preferential web content and services [50].

To realize these intelligent content retrieval models, connectivity is a core feature as users share files, publish articles, comment on others' blogs, view each other's profile and add new members to their networks. These are typical operations in today's online social networks such as Facebook[1], Instagram[2] and Twitter[3]. In these networks, members feed in information every day, resulting in a continuous flow of data. In this myriad of social information production, data classification and clustering can facilitate the process of analyzing and building meaningful infer-

[1]http://www.facebook.com/
[2]http://www.instagram.com/
[3]http://twitter.com/

ences. For example, grouping similar web content could help in finding problems such as detecting copyright violations [51] or building communities. Different approaches, such as clustering and classification [52], are used to achieve the goal of grouping data in social networks. The ability to group users into communities can serve important business needs such as targeted advertisements or enhancing online shopping, through viewing personalized products or services [50]. Allowing more personalization per individual user is achieved through special recommendations for a specific user or through providing page categorizations for users such as Google News[4].

Communities refer to a group of nodes that have high affinities, as revealed by multiple connections between the group members and fewer connections with members in other groups [53]. Figure 2.1 illustrates of a typical community structure containing a network with three groups. Nodes can represent users and edges represent connections between them, like friendships. The interconnections within the groups are high, while the external connections are rare.

Finding communities in social networks can help in disclosing underlying network properties in order to understand and summarize the whole network. Detecting communities can be useful in many real-life fields. For example identifying fraudulent actions in telecommunications networks by recognizing groups of users who have unexpected behaviors in terms of usage [54]. Other examples might relate to grouping pages of related topics on the web or finding papers related to a specific topic in a citation network or simply building communities of practice which gather professionals into common fields of interest. Community detection is also important to identify powerful nodes in the network, based on their structural position to initiate influential campaigns [55].

[4]http://news.google.com/

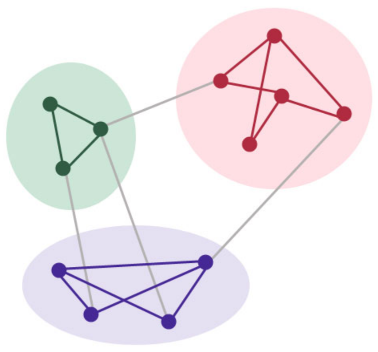


Figure 2.1: A Sample Community Structure

The objective of this chapter is to provide users or organizations with online means of identifying or building communities over the web. We focus on the social web and provide new techniques to detect and build robust communities. The clustering algorithms discussed in this chapter are based on the following categorizations: Hierarchical, Partitional and Density-based algorithms. Each of these approaches features intrinsic techniques such as threshold or centroid techniques. We also discuss and compare six important algorithms used for clustering purpose namely: Link-based (Single-Link, Average-Link and MST Single-Link), K-means, ROCK and DBSCAN algorithms. We propose two complementary algorithms: 1) a Similarity-CNM algorithm which uses the Jaccard Similarity Measure to first infer a virtual network from the original network, resulting in what we label as a 'similarity social network' or simply a 'virtual social network', and 2) an ECD-Jaccard algorithm[5] which assigns weights to edges in the virtual social network and then applies a quality-optimized version of a CNM algorithm[48] to detect communities. This work is based on CNM because it is used in many applications and it has an acceptable speed. The use of network links or connections to measure similarity is more generic than using node attributes like age, interest, etc., which may be context specific. Our techniques show that by pre-processing the original network, we can reduce the time that the CNM algorithm needs to generate communities with a greater structural quality. We propose a pre-processing method that prepares the social network for community detection algorithms. This preparation is shown to

[5]ECD refers to 'Enhancing Community Detection'

enhance the results generated from such algorithms.

The rest of the chapter is organized as follows: Section 2.2 defines clustering and introduces some important terms and concepts related to clustering as well as the different types of clustering techniques. Section 2.3 reveals different clustering algorithms and presents a comparison among candidate clustering algorithms. We also describe a case study related to the use of clustering algorithms in social networks where we evaluate some of the candidate algorithms' clustering performances. In Section 2.4, we introduce and discuss our algorithms to enhance community detection. Also in this section, the quality measures used in community enhancement are explained. Section 2.5 presents an experimental comparison of the proposed algorithms for both artificial and real-world social networks. Section 2.6 includes relevant works in the field of community detection. Finally, Section 2.7 concludes with a summary of results.

2.2 Clustering in Social Web

Dividing people into different groups is human nature. Previously, people used clustering in order to study phenomena and compare them with others based on certain set of rules. Clustering refers to grouping similar things together. It is a division of data into groups of similar objects: each group is called a cluster. Each cluster consists of objects that embody some common similarities and are dissimilar to objects in other groups [56]. We can find many definitions for clustering in the literature [57, 58, 59, 60, 61, 62] but the most common definition is partitioning data into groups (called clusters), based on certain criteria, that the data grouped in one cluster should share. These criteria include common similarities calculated using distance measurements.

We can define clustering in the context of real-world social network by grouping individuals with high friendship relations internally and scattered friendship externally [63]. With clustering, we can identify groups of interest or communities, which share common properties which can be used to study these groups and understand their behavior. Amazon[6] for example provides users with recommendations based on their shopping history. Twitter also recommends new friends (people to follow) to members based on several factors, such as being the friend of a user's friend [64].

Clustering can be used for summarizing large inputs. So instead of applying algorithms on entire data sets, we can reduce the data sets based on specific clustering criteria [52]. Clustering analysis has been used in many research fields such as image analysis, data mining, pattern recognition, information retrieval and machine learning [62]. On the web, identifying groups of data or users would facilitate the availability and accessibility of data. Using clusters in the web is thus an appealing approach to counter the increasing numbers of Internet users and plethora of data, especially in online social networks, where tremendous number of users and data are interlinked.

2.2.1 Cluster Structure

Distance and Similarity Measures

Any clustering algorithm has a similarity factor (proximity matrix) in order to organize similar objects together. It is important to understand measures of similarity. What makes two clusters join? What makes two points similar? And how do we calculate the distance (dissimilarity)?

Rui Xu and Donald Wunsch defined the function of distance or dissimilarity on a dataset X in their survey paper on clustering algorithms [58] by representing an $n*n$ symmetric proximity matrix for a dataset of n elements where the $(i,j)^{th}$ element represents the similarity or dissimilarity measure for the i^{th} and the j^{th} pattern [58].

The family of Minkowski distances is a very common class of distance functions [65] and can be represented as follows:

[6]http://www.amazon.com/

$$D(p_i, p_j) = \sqrt[w]{\sum (p_i - p_j)^w} \tag{2.1}$$

Where w is a parameter with a value greater than or equal to 1. Based on the value of w different distance functions can be represented such as the Hamming distance (w=1), Euclidean distance (w=2) and Tschebyshev distance (w=∞). Other similarity measures are cosine correlation measure and the Jaccard measure [65]. Further discussion can be found in Xu and Wunsch survey paper of clustering algorithms [58].

Dendrogram Data Structure

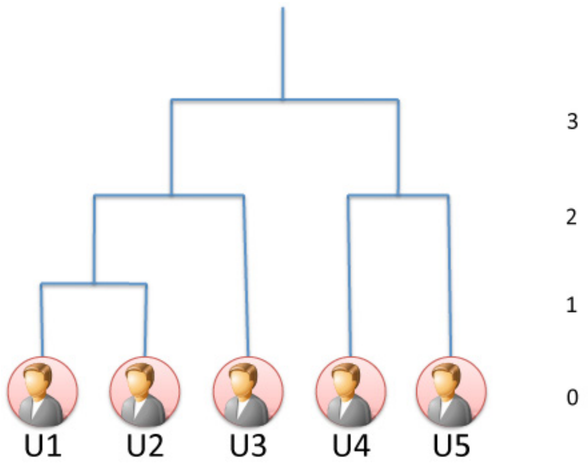


Figure 2.2: Dendrogram Structure

One of the basic structures in the clustering environment is the dendrogram, which is a tree data structure used to form a hierarchical cluster. Figure 2.2 shows a sample dendrogram with four levels. The dendrogram can be represented as a set of triples *S = {[d, k, {. . . }]}* where *d* represents the threshold, *k* is the number of clusters and *{. . . }* is the set of clusters. Figure 2.2 shows a dendrogram for detecting a cluster in a group of five users based on their distance similarities. This dendrogram could be represented by the following set *S = { [0, 5, {{U1}, {U2}, {U3}, {U4}, {U5}}], [1, 4, {{U1,U2}, {U3}, {U4}, {U5}}], [2, 2, {{U1,U2,U3}, {U4,U5}}], [3, 1, {U1,U2,U3,U4,U5}] }* [52]. The dendrogram represents a set of clusters. Most of the algorithms considered in this chapter are hierarchical algorithms.

Proximity Between Clusters

Proximity calculation is the most important step in identifying clusters. It is used to measure how close the data are to each other, and differs according to the algorithm in use. For example, agglomerative hierarchical clustering techniques such as single-link, complete link and group average have different ways to determine the proximity threshold. The single-link defines the proximity as the closest distance between two elements in two different clusters or simply the shortest path between the two nodes in different clusters. Complete link calculates the largest distance between two points in two different clusters or the largest edge between the two nodes in different clusters. In group average the proximity is defined to be the average length distance of all elements from the two different clusters [62]. Figure 2.3 illustrates the three approaches:

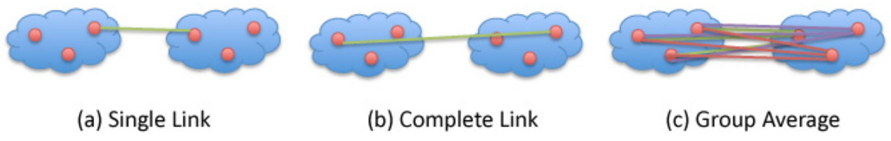


Figure 2.3: Cluster Proximity

2.2.2 Clustering Types

There are many kinds of clustering algorithms available in the literature [57, 58, 61, 56]. They can be categorized based on the cluster structure (hierarchical, partitional), data types and structure (numerical, categorical) or data size (large datasets) [52]. In general, clustering approaches can be divided into four main types: hierarchical, partitional, density-based and meta-search controlled [66]. In this chapter, we will discuss hierarchical, partitional and density-based clustering.

The Hierarchical and Partitional algorithms partition the data into different non-overlapping subsets. A partition of a dataset $X = \{x_1, x_2, ..., x_N\}$, where $x_j = (x_{j1}, x_{j2}, ..., x_{jd}) \in \Re^d$ with each measure x_{ji} called a feature (attribute, dimension or variable) and d is the input space dimensionality [58], is a collection $C = \{C_1, C_2, ..., C_k\}$ of k non-overlapping data subsets. $C_i \neq \oslash$ (non-null clusters) such that $C_1 \cup C_2 \cup \ldots \cup C_k = X$, where X is the super cluster and $C_i \cap C_j = \oslash$ for $i \neq j$ [67]. The data partition is overlapping if the condition ($C_i \cap C_j =$ for $i \neq j$) is ignored and in that case the cluster will have sub clusters of different levels inside it [67].

Hierarchical Clustering

In hierarchical clustering, the clusters are represented as a tree called dendrogram [68]. They can be either top-down (divisive) or bottom-up (agglomerative). Most of these algorithms need a threshold parameter that tells the algorithm when to stop looking for subgroups. Figure 2.4 shows a graphic representation of divisive and agglomerative algorithms.

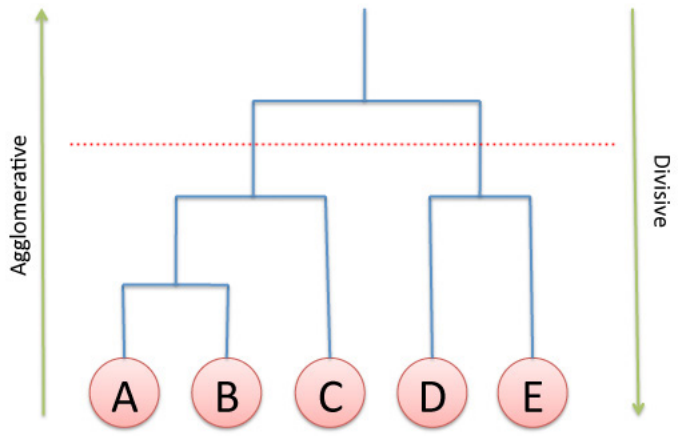


Figure 2.4: A Dendrogram that Represents Divisive vs Agglomerative Clustering. Two Clusters are Generated When Cutting the Dendrogram at a Specific Level

In divisive hierarchical clustering, the algorithm starts from the global cluster that contains all the elements and then the data is divided into sub clusters. We need to find out which clusters to split and how to perform the splitting [65]. While in agglomerative hierarchical clustering, the algorithm starts from a single cluster and then every two clusters are merged together until a global cluster is achieved. DBSCAN is a hierarchical clustering algorithm represented in [69] and used to group articles together that refer to the same event and have similar patterns.

The basic idea behind clustering is to find a distance/similarity measure between any two points such as a Euclidean distance or a cosine distance, etc. In particular, this is the shortest path in linkage algorithms based on linkage metric. To calculate the distance between two points, these algorithms include single-link, average-link and MST link techniques.

Hierarchical algorithms are represented using a proximity matrix (distance matrix) assuming it is symmetric which means that it require the storage of $\frac{1}{2}n^2$ proximities, where n is the number of elements [62]. The total space complexity is $O(n^2)$ and the time required for computing the proximity matrix is $O(n^2)$ [68]. In general, agglomerative hierarchical clustering does not have difficulties in selecting initial points as the algorithm will starts from single clusters. However, they are expensive algorithms in terms of time and space which limit their usage with large scale datasets [58]. We will focus on agglomerative hierarchical algorithms in this section such as Single-Link, Average-Link and MST Single-Link algorithms.

Partitional Clustering

Partitional algorithms have a fixed number of clusters where data is divided into a number of subsets [61]. The most common example is the K-means algorithm that starts by selecting random means for K clusters and assigns each element to its nearest mean. K-means algorithms are $O(tkn)$, where t is the number of iterations [68], k denotes the number of clusters and n the size of the data being clustered. These algorithms use a number of relocation schemes that provide optimization to the clusters, which means the clusters can be refined at each revisited step and thus give an advantage over hierarchical clustering [61].

Density-Based Clustering

In density based algorithms, the cluster is a dense region of data objects. The points density is higher inside the cluster than outside the cluster. It is used most when the shapes of the clusters are irregular and contain noise and outliers [1]. DBSCAN is an example of a density-based algorithm. In the worst case, the time complexity for this algorithm is $O(n^2)$, but in low dimensional spaces the time would be reduced to $O(nlogn)$ [62].

Meta-Search Controlled Clustering

The meta-search controlled clustering approach treats clustering as an optimization problem where a global goal criterion is to be minimized or maximized [1]. Even though these algorithms provide flexibility, their runtime is unacceptably high. Cluster detection can be performed using genetic algorithms or two-phase greedy strategy [1].

In this thesis, we will focus on hierarchical, partitional and density-based algorithms. Next, we will discuss these algorithms in detail.

2.3 Algorithms and Analysis

In this section, we will discuss and compare the following six clustering algorithms:

1. Link-based Algorithms
   (a) The Single-Link Algorithm
   (b) The Average-Link Algorithm
   (c) The Minimum-Spanning-Tree Single-Link Algorithm
2. The K-means Algorithm
3. The Robust Clustering Using Links Algorithm (ROCK)
4. The Density-Based Spatial Clustering of Applications with Noise algorithm (DBSCAN)

2.3.1 Link-based Algorithms

TLink-based algorithms are agglomerative hierarchical algorithms where the dendrogram starts with individual objects and the proximity threshold is set to zero. Then the value of the threshold is increased and based on that value the algorithm checks if two elements should be merged in one cluster or be kept apart. After a number of iterations all the elements will belong to a single super cluster.

The general algorithm for the hierarchical agglomerative algorithms can be described as shown in Algorithm 2.1.

Algorithm 2.1 General Hierarchical Agglomerative Algorithm

```
1. Set the proximity threshold and calculate the proximity matrix.
2. Start with individual clusters.
3. Based on the threshold merge the closest clusters.
4. Update the threshold according to the new clusters.
5. Repeat steps 3 and 4 until all the elements are in one super cluster.
```

The Single-link, Average-link and Minimum-Spanning-Tree algorithms, which are link-based algorithms of the agglomerative hierarchical type will be discussed next.

Single-Link

The single link algorithm is based on the distance between clusters that are connected by at least one edge. First, it calculates the distance between the elements in the clusters. Then the proximity threshold is compared to the minimum distance to determine whether to merge the two clusters or not. The single–link distance between two clusters C_i and C_j can be represented by the following formula [65] :

$$D(C_i, C_j) = min_{x \in C_j, y \in C_i}(x - y) \tag{2.2}$$

The Single-link follows the general approach of linked-based algorithms described in Algorithm 2.1. The time and space complexity of the Single-Link Algorithm is $O(n^2)$ [58]. This complexity is a problem when working with very large data, which is the case when clustering large real web datasets such as social networks.

The single link is sensitive to noise and outliers actually suffer from the chain effect [70]. This effect occurs when a single link algorithm merges two clusters based on two points in these two clusters that are close to each other, regardless of the other points of the clusters that are far away. A single-link does not provide a solution for this problem [52], but algorithms such as ROCK could provide a solution.

Average-Link

The average link algorithm is similar to the single-link algorithm but it uses different techniques to merge two clusters. It uses the average distance between any two points in the two different clusters and checks if it is less than the proximity threshold in order to merge the clusters.

As with a single-link algorithm, we start with individual clusters and merge them until one cluster is formed, but unlike the single link the distance of all pairs of points between the two different clusters need to be calculated.

The average distance between two clusters C_i and C_j could be represented by the following formula:

$$D(C_i, C_j) = \frac{\sum_{x \in C_j, y \in C_i} (x - y)}{C_i . C_j} \tag{2.3}$$

The time and space complexity of the Average-Link Algorithm is $O(n^2)$ [58]. Which is similar to single-link algorithms so it has the same problem.

Minimum-Spanning-Tree Single-Link

In this approach, a minimum spanning tree connects all the elements of a given set in a way that minimizes the sum of the adjacency values for the connected elements [71]. The MST single link algorithm is a combination between single link algorithms and minimum spanning tree.

The Prim-Jarnik algorithm [71] is used in this approach for the minimum spanning tree with single technique. This algorithm builds the minimum spanning tree starting from a single cluster (root) as expressed in Algorithm 2.2.

Algorithm 2.2 MST Single-Link Algorithm

1. Mark all elements of the graph as not visited.
2. Choose any element you like as the root and mark it visited (cluster C created).
3. The smallest-weight edge e= (v, u) that connects one vertex v inside the clustering C is chosen and added to the spanning tree T.
4. Repeat until all vertices are visited and the minimum spanning tree is formed.

The time and space complexity of the MST Single-Link Algorithm is $O(n^2)$ [52]. This is similar to the single-link and average-link algorithms. The MST Single-Link Algorithms results in fewer clusters than the single link algorithm because the proximity circles do not expand as much as in the single link.

2.3.2 *K*-means Algorithm

K-means is a partitional algorithm. It uses the idea of a centroid, which is the mean of a group of points. It has high performance characteristics and it is one of the oldest and most used clustering algorithms. Figure 2.5 illustrates the idea of centroid.

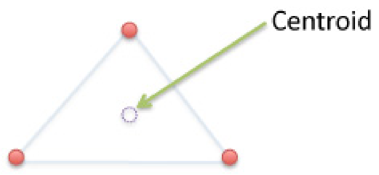


Figure 2.5: The Centroid Approach

The *K*-means algorithm starts by choosing the *K* initial centroids. The simplest approach is to choose random centroids. Then the points are assigned to their closest centroid to form *K* clusters [72]. Depending on the points assigned to the cluster the centroid position is updated. We repeat the update until there are no more points to add or the centroids remain unchanged. The *K*-means algorithm can be represented as shown in Algorithm 2.3.

Algorithm 2.3 K-means Algorithm

```
1. Select $K$ points as initial centroids.
2. Form $K$ clusters by assigning each point to its closest centroid.
3. Re-compute the centroid of each cluster.
4. Repeat steps 2 and 3 until centroids are unchanged.
```

The K-means is fast compared to other clustering algorithms. Its computational time is $O(tkn)$ where t is the number of iterations, k represents the number of clusters and n is the number of data points we want to cluster. The space complexity for K-means is $O(n)$ which is much better than link based algorithms.

Different runs of the K-means will produce different results since we randomly initialize the centroids. This will produce poor clustering results. So choosing the right initial centroids is very important in order to create good quality clusters [73]. It is better to choose centroids in regions with high concentration of data points as proposed by David Arthur and Sergei Vassilvitskii in their K-mean++ article [74].

The K-mean is efficient for large datasets [73] and works well with numerical data. But a challenge occurs when it is used with categorical data such as strings since we need to find a good way to represent nonnumeric values in a numerical way.

2.3.3 Robust Clustering Using Links (ROCK)

ROCK is an agglomerative hierarchical algorithm. It uses links as a similarity measure rather than measures based on distance. It clusters points that have many common links. As an agglomerative hierarchical algorithm it starts from single clusters and merges these clusters until a super single cluster is formed. For the ROCK algorithm we need to define a minimum number of clusters that we want to form in order to stop the algorithm before all the elements are grouped into one single cluster.

Goodness Measure

In the process of merging clusters in the ROCK algorithm, we need to determine the best pair of clusters to merge together. Thus a goodness measure is used. Actually for ROCK algorithms the best clusters are those that maximize the goodness measure. The goodness measure for two clusters C_i and C_j is represented as follows [75]:

$$g(C_i, C_j) = \frac{link[C_i, C_j]}{(n_i + n_j)^{1+2f(\theta)} - n_i^{1+2f(\theta)} - n_j^{1+2f(\theta)}} \tag{2.4}$$

where $link[C_i, C_j]$ represents the number of cross links between clusters C_i and C_j that is

$$\sum_{p_q \in C_i, p_r \in C_j} link(p_q, p_r) \tag{2.5}$$

There is a link between two data points if a common neighbor exists between them. For the ROCK algorithm to merge two clusters the focus will be on the number of links n_i, n_j between all paired points of the two clusters C_i, C_j. A large number of links should indicate a higher probability that the two points belong to the same cluster and give the best cluster.

The denominator of the goodness measure is a normalization process that estimates the expected number of links between pairs of points each from different clusters. Where $(n_i + n_j)^{1+2f(\theta)}$ is the number of links between pairs of points in the merged cluster (after merging the two clusters) and $n_i^{1+2f(\theta)}$ is the expected number of links between points within the cluster i (the number of links in the cluster before merging) and $n_j^{1+2f(\theta)}$ is the expected number of links between points within the cluster j (the number of links in the cluster before merging). This property of the goodness measure prohibits data points that have few links between them from being assigned to the same cluster [75].

Any pairs of clusters that will maximize the goodness measure will be the best pairs to merge. With algorithms that are based on similarity distance only, it will be difficult to determine if two clusters are separate because this kind of measurement may merge two clusters if there are two points close together even though these points do not have large number of common neighbors [68]. Thus the ROCK algorithm uses links as its name implies.

ROCK Algorithm

The ROCK algorithm needs the following arguments:

1. The set of points that we want to cluster
2. The minimum number of clusters to have to stop the ROCK algorithm before all points are merged in one cluster
3. Proximity that is required between two points in order to form a link between them

The ROCK algorithm could be expressed as shown in Algorithm 2.4.

Algorithm 2.4 ROCK Algorithm

```
1. Create a cluster for each point.
2. Use a goodness measure to evaluate if two clusters should be merged or not
   (the best are the ones that maximize the value of the goodness measure).
3. Repeat step 2 until the number of clusters formed is equal to the minimum
   number required to stop or the number of cluster does not change between
   iterations.
```

The space complexity of the ROCK algorithm is $O(n^2)$ [52], while the time complexity is $O(n^2 logn)$ [75].

The ROCK algorithm is best used with categorical data such as keywords, Boolean attributes that use the Jaccard coefficient to measure similarity [73]. It works well on large data sets. One of the advantages of using the ROCK algorithm is its ability to handle outliers effectively. Outliers are points that lies in a far distance from the other points. Which means these points can be easily discarded, as they will not participate in the clustering process [75].

2.3.4 Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

The DBSCAN algorithm is a density-based algorithm that uses density as a measurement other than links or distance between points.

Density-based algorithms are based on using density to identify the boundaries of objects. So clusters are identified based on their points density within a specific region. Figure 2.6 explains this concept where we can identify three clusters in the figure. The points that don't belong to the clusters are identified as noise and DBSCAN is used to discover clusters and noise in a dataset.

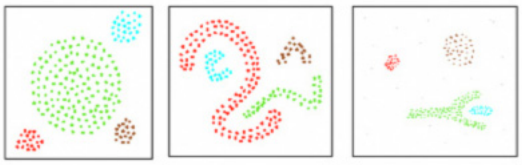


Figure 2.6: Density-based Clustering

The DBSCAN can be described as follows (Figure 2.7): any two core points should be put in the same cluster if they are close to each other within a distance of *ε(Eps)* [68]. Where *ε(Eps*), stands for epsilon and is a value that helps to define an epsilon neighborhood for any given data point *p* [1].

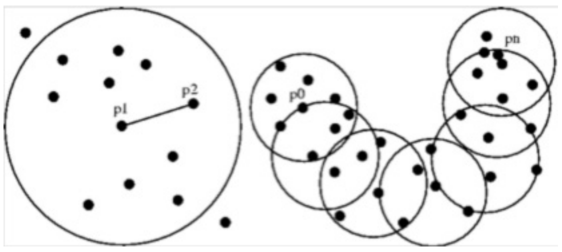


Figure 2.7: DBSCAN Core Points, Border Points and Noise [1]

To understand the concept of center points let us look at Figure 2.7. The large circles are the epsilon neighborhood for points *p* and *q*. Each of them is a center of one of the circles. The circle radius is ε and *minPoints* represents the minimum number of points that must be inside the circle for a data point to be considered a core point. The points that are on the border of the cluster are called border points. A point p_1 is directly density-reachable from a data point p_2 with respect to ε and *minPoints* if there is a list of points $p_1, ..., p_n$, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly density-reachable from p_i [1]. The following two conditions should be met:

1. p_1 inside the epsilon neighbor of p_2
2. There are more than *minPoints* data points inside the epsilon neighborhood of p_2

The DBSCAN algorithm is expressed in Algorithm 2.5.

Algorithm 2.5 DBSCAN Algorithm

```
1. Define the points as core, border, or noise points.
2. Eliminate noise points.
3. Put an edge between all core points that are within $Eps$ of each other.
4. Make each group of connected core points into a separate cluster.
5. Assign each border point to one of the clusters of associated core points.
```

The time complexity for the DBSCAN algorithm is $O(n^2)$ [76], where n is the number of points. DBSCAN can handle noise and different shape clusters because it is based on density. It can discover many clusters that are not found by the K-means algorithm. But this algorithm will have problems with clusters of very different densities as the algorithm requires that the object's neighbors have high density [58] with high-dimensional data [62]. The DBSCAN uses R*-tree in order to improve the determining of the points within a ε distance [73]. R*-tree will reduce the time complexity of the DBSCAN to $O(nlogn)$ [1].

2.3.5 Discussion and Evaluation

In this section, we have discussed the complexity of clustering algorithms and other related issues. There are many criteria that decide the use of one algorithm over others. The two main criteria are time complexity and if they handle data with high dimensionality.

Scalability Analysis

To deal with a large number of elements we need to evaluate the computational complexity of the algorithm under consideration, in other words how long does this algorithm take to construct the cluster?. There is a big difference between clustering groups of people on Facebook with millions of registered users and clustering a local newsgroup of some hundred users. To understand how crucial the data size is we need to understand how each algorithm deals with memory size (space complexity) and the number of operations performed to cluster a set of data (time complexity). Table 2.1 shows both of these metrics for the algorithms discussed. Here k denotes the number of clusters, t the number of iterations, and n the size of the data being clustered. It is obvious that the problem is with the $O(n^2)$ algorithms specially when n is large. In [58] Rui Xu and Li Wunsch compared the time and space complexities of these algorithms and provided additional algorithms that can handle very large data sets such as CLARA, CLARANS and BIRCH.

Algorithm name	Space complexity	Time complexity
Single-Link	$O(n^2)$	$O(kn^2)$
Average-Link	$O(n^2)$	$O(kn^2)$
MST Single-Link	$O(n^2)$	$O(n^2)$
K-means	$O(n+k)$	$O(tkn)$
ROCK	$O(n^2)$	$O(n^2 log(n))$
DBSCAN	$O(n^2)$	$O(n^2)$ or $O(nlog(n))$ with R*-tree

Table 2.1: Space and Time Complexities for Clustering Algorithms

It is obvious that hierarchical clustering algorithms are not suitable for large datasets because of their complexities. The K-means is the most efficient algorithm among them as the complexity is almost linear [58], but it cannot handle categorical data which is very important when clustering the web. DBSCAN can be improved by using spatial indices on data points such as R*-tree that will reduce the time complexity from $O(n^2)$ to $O(nlog(n))$ and generates more efficient queries

[58]. It is important to mention that indexing spatial data faces difficulties in high dimensions and this subject is an active area of research [52].

Dimensionality Issues

The world that we deal with is three dimensional and if we want to cluster worlds of higher dimensionalities we need to know that these worlds are governed by different rules and different proximities [52]. Actually higher dimensionality means larger computations which will slow the algorithm down.

High dimensionality produces a problem in data separation as the distance between the point and its nearest neighbor has no difference than the distance from that point to other points when the dimensionality is high [68]. The 'curse of dimensionality' is a problem that is related to high dimensionality. The term was introduced by Bellman to indicate the exponential growth of complexity in a high dimensionality situation [58], which indicates that the distance between any set of points in high dimensions are the same. In such situation there will be no effect for clustering algorithms that are based on distance measurements. Aggarwal provided a solution to this problem [77].

Case Study and Evaluation

In this section, a case study is used to further explain clustering algorithms. The scripting language is explained and also how the environment was set up to run the algorithms and obtain results. Also we will discuss the results obtained using each algorithm.

The issue of identifying articles of similar topic is of great potential in the intelligent web environment. In our case study we will use clustering algorithms to help in grouping similar articles. Data was collected from Delicious.com, which is a social bookmarking service that allows users to share, store and discover web bookmarks [78]. Since we are dealing with categorical data and keywords, represented by articles titles, we will use ROCK and DBSCAN algorithms to define different clusters and group similar titles together. The codes for both algorithms are available [52].

Data Collection

For the two experiments, we have collected a list of 48 titles for different articles from Delicious.com and saved them in a CSV file. Each title was assigned a unique ID and a username for the person who bookmarked that title. Two or more users can bookmark the same title. A sample of the dataset (11 out of 48 titles) is illustrated in Table 2.1.

ID	Name	Title
776	user01	Google Sites to Add Social Networking in ‘Layers’
774	user01	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
740	user01	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
770	user01	Twitter is NOT a Social Network, Says Twitter Exec
722	user01	An Open Source Collaborative Network
744	user02	Google Sites to Add Social Networking in ‘Layers’
710	user02	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
730	user03	Google Sites to Add Social Networking in ‘Layers’
777	user03	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
756	user03	An Open Source Collaborative Network
733	user03	How To Discover Your Money Making Niche

Table 2.2: Sample Dataset Collected from Delicious.com

Evaluation and Discussion

The two algorithms are implemented in Java language and to execute and debug them we used BeanShell, which is a free Java interpreter. The code commands were executed through the command line on Windows OS environment.

ROCK Algorithm

First we used the ROCK Algorithm from [52] to cluster the dataset. The algorithm uses the Jaccard Coefficient to measure the similarities between different titles. It compares the common terms or keywords in the titles and based on their similarities are grouped together.

To start the experiment we have loaded Delicious.com titles using the 15 most common terms and stored these titles in an array. The ROCK algorithm was used to cluster the dataset with a minimum number of clusters equal to 5. This parameter will allow the ROCK algorithm to stop before grouping all the data in one cluster. The threshold of 0.2 is used to represent the required proximity between two points that can be linked. Algorithm 2.6 represents the code used to execute the algorithm and print the result.

Algorithm 2.6 ROCK Algorithm Execution Code

```
1. DeliciousDataset ds = DeliciousData.createDataset(15);
2. DataPoint[] dps = ds.getData();
3. ROCKAlgorithm rock = new ROCKAlgorithm(dps, 5, 0.2);
4. Dendrogram dnd = rock.cluster();
5. dnd.print(16);
```

The results of our experiment for the ROCK algorithms at level 16 shows 8 clusters, (Table 2.3). We noticed that the algorithm clustered similar titles such as title ID 799 and title ID 688 together. On the other hand, there are articles with similar titles, but the algorithm did not merge them in one cluster such as title ID 520 that is in cluster 4, and title ID 681 in cluster 7. The algorithm also defined non-obvious clusters such as the titles in cluster 4; which contained different titles that are grouped together because they contain similar terms related to ‘social network’ topics. The ROCK algorithm will compare titles based on keywords in these titles.

Clusters for: level-16, Goodness = 1.973889261532508		
Cluster No.	ID	Title
1	799	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
1	688	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
2	708	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
2	774	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
2	710	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
3	722	An Open Source Collaborative Network
3	715	An Open Source Collaborative Network
4	520	Twitter is NOT a Social Network, Says Twitter Exec
4	566	Twitter is NOT a Social Network, Says Twitter Exec
4	744	Google Sites to Add Social Networking in ‘Layers’
4	730	Google Sites to Add Social Networking in ‘Layers’
4	776	Google Sites to Add Social Networking in ‘Layers’
4	770	Twitter is NOT a Social Network, Says Twitter Exec
5	740	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
5	720	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
6	777	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
6	795	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
7	681	Twitter is NOT a Social Network, Says Twitter Exec
7	500	Twitter is NOT a Social Network, Says Twitter Exec
7	790	Twitter is NOT a Social Network, Says Twitter Exec
7	780	Google Sites to Add Social Networking in ‘Layers’
8	735	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
8	726	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials

Table 2.3: ROCK Algorithm Results

DBSCAN Algorithm

We applied the DBSCAN algorithm to the same dataset. The algorithm also used the Jaccard Coefficient to measure the similarities between different titles. To start the experiment we have loaded Delicious.com titles using the 15 most common terms only and stored these titles in an array. A cosine distance is used as a distance metric. The DBSCAN algorithm invoked to cluster the dataset with a *distance metric*, *ε (neighbor threshold)*, *minPoints* and *the term frequency*. Algorithm 2.7 represents the code used to execute the algorithm and print the result.

Algorithm 2.7 DBSCAN Algorithm Execution Code

```
DeliciousDataset ds = DeliciousData.createDataset(15);
DataPoint[] dps = ds.getData();
CosineDistance cosD = new CosineDistance();
DBSCANAlgorithm dbscan = new DBSCANAlgorithm(dps, cosD, 0.7, 2, true);
dbscan.cluster();
```

The results of our experiment for the DBSCAN algorithms are shown in Table 2.5. The results were more accurate than the ROCK algorithm results. All similar titles are clustered together such as clusters 2 and 3 and the titles are exactly the same as each other. Non-similar titles in cluster 1 (title ID 776, title ID 566) and cluster 4 (title ID 722, title ID 711) were also defined by the algorithm, In cluster 1 the titles are grouped based on the keyword ‘social networks’ and cluster 4 grouped all the titles related to the ‘open source’ topic. The algorithm was also able to recognize noise elements where these points did not belong to any cluster.

1	790	Twitter is NOT a Social Network, Says Twitter Exec
DBSCAN Clustering with NeighborThreshold = 0.7 minPoints = 2		
Cluster No.	ID	Title
1	776	Google Sites to Add Social Networking in ‘Layers’

1	790	Twitter is NOT a Social Network, Says Twitter Exec
1	566	Twitter is NOT a Social Network, Says Twitter Exec
1	744	Google Sites to Add Social Networking in ‘Layers’
1	780	Google Sites to Add Social Networking in ‘Layers’
1	500	Twitter is NOT a Social Network, Says Twitter Exec
1	730	Google Sites to Add Social Networking in ‘Layers’
1	770	Twitter is NOT a Social Network, Says Twitter Exec
1	681	Twitter is NOT a Social Network, Says Twitter Exec
1	520	Twitter is NOT a Social Network, Says Twitter Exec
2	774	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
2	708	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
2	777	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
2	726	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
2	795	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
2	735	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
2	710	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
3	740	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
3	530	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog

1	790	Twitter is NOT a Social Network, Says Twitter Exec
3	688	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
3	685	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
3	799	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
3	720	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
4	722	An Open Source Collaborative Network
4	590	An Open Source Collaborative Network
4	711	XWiki - Open Source Wiki and Content-Oriented Application Platform
4	600	An Open Source Collaborative Network
4	715	An Open Source Collaborative Network
4	756	An Open Source Collaborative Network
5	690	Apple: Sorry, Steve Jobs Isn't a Ninja
5	736	Apple: Sorry, Steve Jobs Isn't a Ninja
6	499	How To Discover Your Money Making Niche
6	733	How To Discover Your Money Making Niche
7	743	How To Create WordPress Themes From Scratch Part 1
7	533	How To Create WordPress Themes From Scratch Part 3b
7	694	How To Create WordPress Themes From Scratch Part 2

1	790	Twitter is NOT a Social Network, Says Twitter Exec
7	510	How To Create WordPress Themes From Scratch Part 3a
Noise	540	iPhone SDK 3.0 Playing with Map Kit - ObjectGraph Blog
Noise	742	This Is The Second Time A Google Engineer Has Been Fired For Accessing User Data
Noise	577	Enhance Your Web Forms with New HTML5 Features
Noise	745	Resize or Move a Complete Flash Animation in One Go
Noise	746	10 Things You Didn't Know About the New #Twitter /via @gigaom #news #sm
Noise	732	How To Handle Customers During Virtual Assistant Problems
Noise	705	Article on Social Media Ad Campaigns
Noise	587	The Business Plan
Noise	791	CSS Color Names
Noise	601	Typography : Web Style Guide 3
Noise	753	in 4 U.S. Adults Now Use Mobile Apps [STATS]

Table 2.5: DBSCAN Algorithm Results

Discussion

From both experiments, the DBSCAN and the ROCK algorithms produce good clustering results for the dataset. We noticed that the DBSCAN advanced the ROCK algorithm to find the correct clusters and outliers as shown in Table 2.5. The ROCK is based on measuring similarity between two clusters as it finds the common neighbors for the two clusters. But relaying on similarity might make the algorithm merge two clusters even if they contain outliers or noise.

2.4 Similarity-Based Community Detection

Basic clustering methods face challenges when dealing with social networks data clustering. Many of the proposed clustering algorithms such as hierarchical clustering and K-Means use distance matrices to build clusters. But social networks data needs algorithms that directly use graph properties such as edge and node betweenness to detect communities. Different community detection algorithms are proposed in the literature such as hierarchical divisive algorithms using shortest-path betweenness [79]. This algorithm starts from the global cluster of the network and iteratively identifies the shortest path of edges that lie between clusters and then removes them to generate cohesive communities. Another popular approach proceeds with clustering based on network modularity [80]. This uses a modularity function to measure the quality of partitioning a network into communities.

In this section, two community detection proposals are suggested. They semantically enrich the network prior to the community detection process, in order to enhance community discovery and deliver a better community structure. This is achieved when the community structure has nodes that are densely connected internally in a network. The complexity of the proposed offline pre-manipulations is $O(n^2)$. This manipulation is done offline to avoid any extra costs. The resulting network is then used by CNM to detect the final communities. In the proposed approach, we use unweighted directed networks and generate the same networks with added edge-weights based on the nodes' similarities. Using this similarity helps in measuring the strength of relationships between nodes, which in turn builds tighter communities based on this relationship. We postulate that this approach enhances the community structure. Our experimental evaluation of artificial and real-world networks supports this claim, based on modularity and normalized mutual information measures, as described earlier.

To detect, find and measure the strength of a community structure in social networks, we use the CNM community analysis algorithm proposed by Clauset, Newman, and Moore [48]. The CNM algorithm is a bottom-up agglomerative clustering method that uses greedy techniques to combine and merge clusters in a network. It is efficient at identifying communities since its running time is $O(nlog^2n)$. We used this algorithm to calculate the modularity (Q) for social networks, where Q states the quality of graph partitioning or the quality of clustering. The modularity (Q) formula is defined as follows [80]:

$$Q = \sum_i (e_{ii} - a_i^2) \tag{2.6}$$

Where e_{ij} is the fraction of edges that connect vertices in group i to vertices in group j and $a_i = \sum_j e_{ij}$. Modularity is based on finding the difference between the number of edges within the communities and the expected number of edges (edges are generated randomly). When the difference is large, we get a better community structure. Values of Q above 0.3 means a significant community structure in a network [48].

High modularity is not always associated with the best partitions. Sometimes high modularity partitions are not optimal [81]. In this chapter, we used the normalized mutual information proposed by Danon, et al. [82] to determine how similar the original communities are to the ones found by the proposed algorithms in this thesis. The value of the mutual information is between 0 and 1 inclusive. If there is an exact match between the found and real communities then the value of the mutual information is equal to 1. If there is less of a match the value of the mutual information decreases. We used this measure to evaluate the accuracy of the results obtained on the benchmark networks. The normalized mutual information is defined as follows [82]:

$$I(A,B) = \frac{-2\sum_{i=1}^{c_A}\sum_{j=1}^{c_B} N_{ij} log(N_{ij}N/N_{i.}N_{.j})}{\sum_{i=1}^{c_A} N_{i.} log(N_{i.}/N) + \sum_{j=1}^{c_B} N_{.j} log(N_{.j}/N)} \tag{2.7}$$

Where A represents the real communities and B represents the detected communities. While c_A and c_B are the number of communities in A and B respectively. In this formula, N is the confusion matrix with rows representing the original communities and columns representing the founded communities. The value of N_{ij} means the number of common nodes that are in the original community i and the founded community j. The sum over the ith row is identified as $N_{i.}$ and the sum over the jth column is identified as N_j.

2.4.1 Similarity-CNM Algorithm for Unweighted Social Networks

The CNM algorithm starts from a single/separate nodes with no edge connections, low modularity and low community structure. It then proceeds to add edges, build communities and merge pairs that increase the modularity. This process tends to quickly build larger communities of low degree nodes, which results in poor maximum modularity values [55]. To solve this issue, we propose to feed the CNM algorithm with an initial set of similar nodes. This initial set of nodes composes of a synthetic virtual social network which is generated from an original one. This similarity pre-processing step helps in building more connected nodes and shaping the structure of the communities more quickly. After this pre-processing step, the CNM algorithm is applied to the resulting connected nodes. Our goal is to enhance the community structure and maximize modularity.

Algorithm 2.8 Similarity-CNM Algorithm

```
1. Start from single nodes and the original social network.
2. Start generating the similarity social network from the original social network.
   (a) For each node $a$ and $b$ do:
       i. Calculate similarity based on Equation 2.8.
       ii. For each node $a$ find the highest similar node $b$.
           • Establish a link between nodes $a$ and $b$.
       iii. Apply CNM algorithm to calculate the modularity ($Q$).
```

The similarity-CNM Algorithm shown in Algorithm 2.8 starts by building a new synthetic social network based on the original social network. The synthetic social network is a virtual social network labeled ‘similarity social network’. During this process, nodes are grouped together to create a virtual social network based on high similarity. If two nodes a and b are highly similar, then a link is virtually established between them. The synthetic link is virtually established in the synthetic network, but no actual link is created in the original social network. At the end of the process, a new virtual network is generated where nodes similarity is derived from the Jaccard Measure [83], as shown below:

$$Similarity(a,b) = \frac{adj_{ab} + cn_{ab}}{n_a + n_b} \tag{2.8}$$

In Equation 2.8, adj_{ab} which represents the intersection of row a and column b in the adjacency matrix, is equal to 1 if there is an edge between nodes a and b and 0 otherwise, cn_{ab} are the number of common neighbors of nodes a and b, n_a and n_b are the total neighbors of nodes a and b respectively.

The original CNM algorithm [48] is applied on the generated synthetic network to discover new communities with better structures. The generated synthetic network includes the original network. That is because the method includes adjacent connections between any two nodes adj_{ab}. This expectation is facilitated by the pre-processing step, which synthesizes new similarities in the virtual network.

This step also favors the generation of communities which are densely connected as shown later in the experiments and performance analysis section of this chapter.

2.4.2 ECD-Jaccard Algorithm for Weighted Social Networks

CNM is usually solicited for unweighted social networks. In our next algorithm, we use the weighted version of CNM to show that by assigning weights to the edges of the network we can infer sets of highly connected nodes. To assign a weight to each edge, we used a simple similarity measure called the Jaccard Similarity [83].

$$JaccardSimilarity(a,b) = \frac{|n_a \cap n_b|}{|n_a \cup n_b|} \tag{2.9}$$

In the worst case, computing the Jaccard similarity occurs in $O(n^2)$ time. So for large data sets, the computations might be expensive, especially that the Jaccard method needs to perform the calculation from the start for each pair, and it does not use any previously calculated information [83]. We note here that this computation was performed offline and before applying the community detection algorithm, which will not reduce the performance of the CNM algorithm. In the case of real-world social networks, the connections and relations are changed dynamically. In that case, our algorithm is able to compute the new relations built between any two nodes in the social network without affecting the performance of the CNM algorithm. The basic steps of our algorithm are shown in Algorithm 2.9.

Algorithm 2.9 ECD-Jaccard Algorithm

```
Input:
$G$, unweighted directed graph
Output:
$G_w$, weighted directed graph where the edges weights are calculated based on nodes
of Jaccard Similarity
$Q$s, modularity values
begin
  //initialize the weighted graph matrix ($G_w$) to zeros
  $G_w = 0$;
  //$a$ and $b$ are nodes in $G$
  //$E_G$ is the set of edges for graph $G$
  for every $(a, b) \in E_G$ do
    //calculate weights using the Jaccard Similarity Measure
    //then assign weights to their appropriate location in $G_w$
    $G_w[a, b] = JaccardSimilarity(a, b)$;
  end for
  Apply CNM algorithm on $G_w$ and generate $Q$s, where
    Q = (number of internal community edges) - (expected number of such edges)
end
```

2.5 Experiments and Performance Analysis

We mentioned earlier that the proposed approaches perform well in detecting communities in online social networks. To illustrate this potential outcome, this section shows our experimental results for both synthetic and popular real-world datasets. We used the modularity Q and the normalize mutual information discussed earlier in our experiments as evaluation metrics to show the performance and accuracy of our algorithms. We implemented the algorithms using Matlab and C++. The maximum modularity (Q) was estimated using the CNM algorithm provided by Clauset, et al. [48]. We also calculated the normalize mutual information using the formula provided Danon, et al.[82]. We performed the experiments on an Apple iMac with Mac OS X version 10.6.8, processor 2.66 GHz intel Core i5 and 4GB memory.

2.5.1 Benchmark Networks Simulation for Similarity-CNM Algorithm

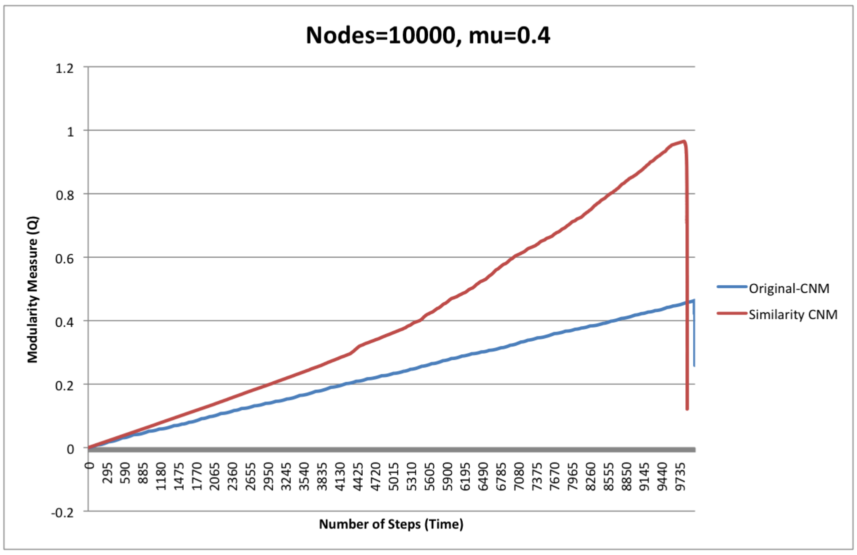
In this experiment, we used benchmark generated networks. There are many generators for testing networks and we used the benchmark network generator developed by Lancichinetti [84]. The generator requires different parameters to produce a directed unweighted network. The values we used to generate different benchmark networks for a range of μ values (varying from 0 to 1 and defined next) are :

$$N = 10000, k = 20, maxk = 50, \mu = 0, 0.1, 0.2, \ldots, 1, minc = 100, maxc = 300$$

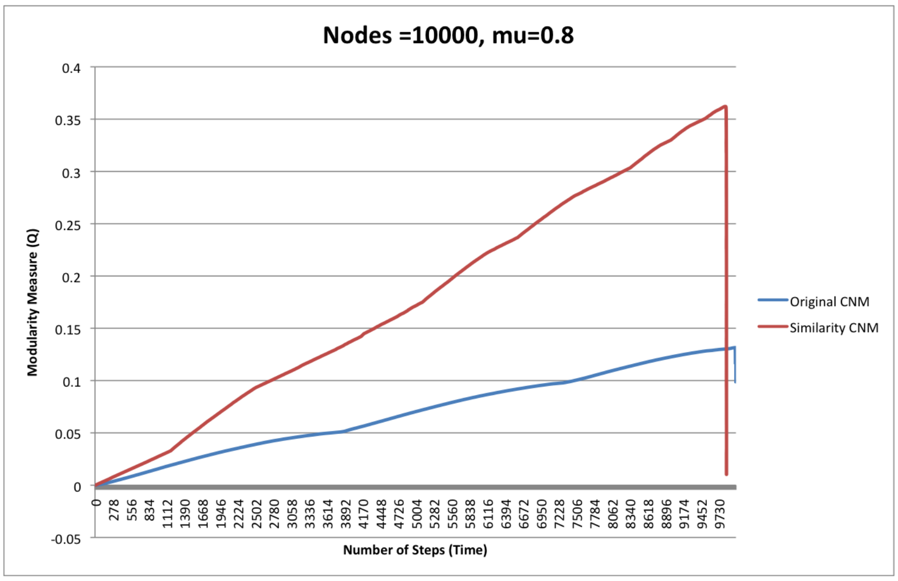
Where (N) is the number of nodes in the network, which we set to 10000 nodes to be equivalent to the number of nodes in the real-world network Flickr that we will evaluate in Section 2.5.2, (k) is the average in-degree for the nodes, ($maxk$) is the maximum in-degree for the nodes, (μ) is the fraction of links a node shares with other nodes in other communities; called also the mixing parameter; ($minc$) is the minimum community size and ($maxc$) is the maximum community size.

We generated six unweighted benchmark networks and performed two experiments on these networks. First the CNM is applied directly on the benchmark networks to generate original communities. Secondly, we applied the Similarity-CNM technique on the benchmark networks followed by applying the CNM algorithm. Then we compared the results generated from applying the CNM on both experiments. Figure 2.8 shows how modularity (Q) evolved over time for networks with μ value of 0.4 and 0.8 respectively. We noticed that Q in general starts small then increased over time to reach its maximum point. Beyond this point, Q value drops sharply as the randomly generated edges exceeds the actual ones as shown in Equation 2.6. By calculating the nodes similarity we noticed higher Q values and better maximum Qs after applying Similarity-CNM technique. The values of maximum Q in synthetic networks outperforms the maximum Q values in original network by more than 25% when μ value is equal to 0.6. Adding a similarity feature to the network edges helps in grouping nodes in robust sets which ensure better community structure.

Table 2.6 and Figure 2.9 show the values of Q_{max} for benchmark networks. For these networks with μ value 0.4 and 0.8, the maximum modularity value for Similarity-CNM outperforms the original CNM algorithm by more than 50% and 23% respectively. Table 2.7 and Figure 2.10 also show that number of steps to reach Q_{max} is better in Similarity-CNM than the original CNM algorithm. For benchmark networks with μ values of 0.4 and 0.8, the number of steps when running the original CNM is 9993 and 9994 steps respectively, while in Similarity-CNM it is reduced to 9819 and 9881 steps respectively.



(a) Maximum Modularity When $\mu = 0.4$



(b) Maximum Modularity When $\mu = 0.8$

Figure 2.8: Maximum Modularity For Different Mixing Parameter

	$\mu = 0$	$\mu = 0.2$	$\mu = 0.4$	$\mu = 0.6$	$\mu = 0.8$	$\mu = 1$
Original CNM	0.981127	0.71149	0.463902	0.243267	0.131554	0.128823
Similarity-CNM	0.984779	0.98204	0.965201	0.49581	0.362154	0.374377

Table 2.6: Comparing Maximum Modularity using CNM Algorithm and Similarity-CNM Algorithm for Benchmark Networks

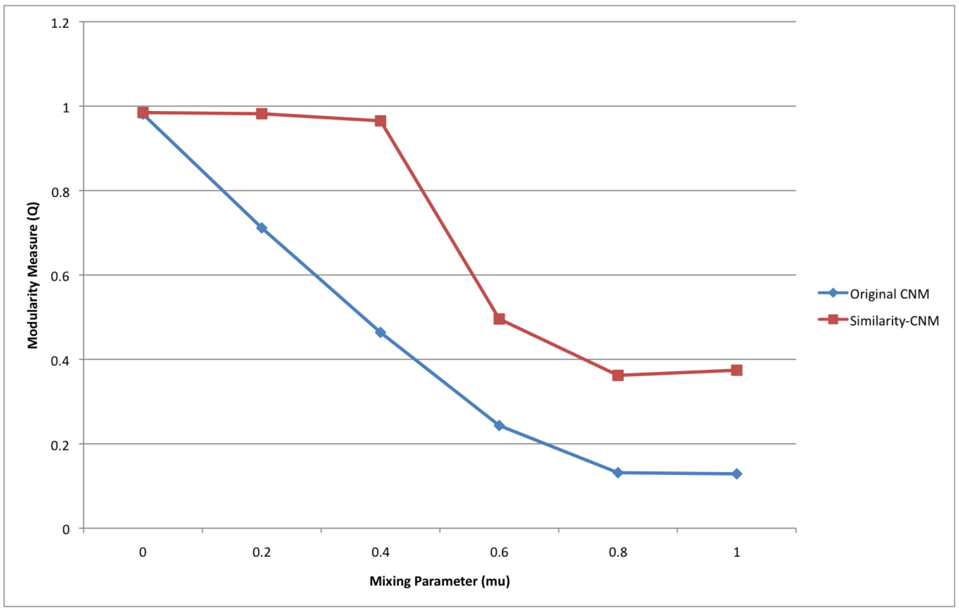


Figure 2.9: Maximum Modularity using CNM Algorithm and Similarity-CNM Algorithm for Benchmark Networks

	$\mu = 0$	$\mu = 0.2$	$\mu = 0.4$	$\mu = 0.6$	$\mu = 0.8$	$\mu = 1$
Original CNM	9943	9987	9993	9996	9994	9993
Similarity-CNM	9388	9674	9819	9881	9832	9783

Table 2.7: Comparing Number of Steps to Reach Q_{max} using CNM Algorithm and Similarity-CNM Algorithm for Benchmark Networks

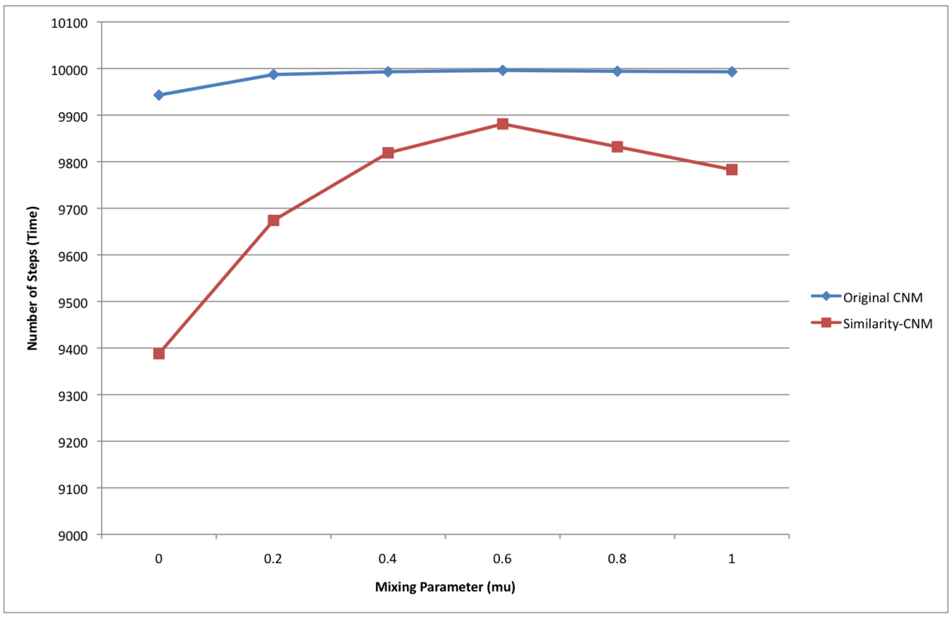


Figure 2.10: Number of Steps to Reach Q_{max} using CNM Algorithm and Similarity-CNM Algorithm for Benchmark Networks

We also used the normalized mutual information measure on the benchmark data to evaluate the accuracy of the benchmark networks results. Normalized mutual information measure estimates the common information between the original and the detected communities. This will show how much the communities that our algorithms generate match the actual ones. Figure 2.11 shows that the Similarity-CNM algorithm detects communities that are 90% similar to the original ones when $\mu = 0$. With that value of μ, the Similarity-CNM algorithm will detect 612 communities compared to only 57 communities found by the CNM algorithm in the original benchmark network as shown in Table 2.8. When $\mu = 0.4$ & 0.8, the communities detected by the Similarity-CNM are 55% and thus almost 2% higher.

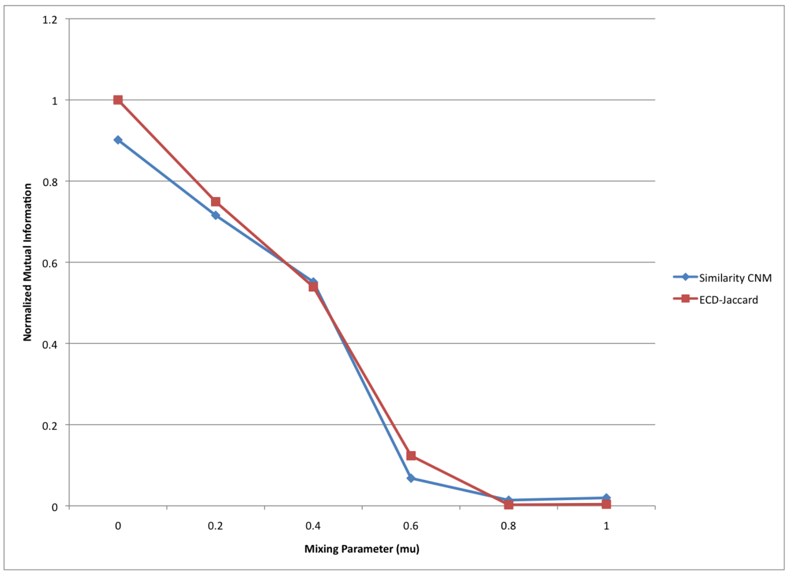


Figure 2.11: Normalized Mutual Information for Different Mixing Parameter Values

μ	CNM	Similarity-CNM	ECD-Jaccard
0	57	612	57
0.2	13	326	56
0.4	7	181	49
0.6	4	119	19
0.8	6	168	6
1	7	217	8

Table 2.8: Number of Communities Detected by the CNM, Similarity-CNM and ECD-Jaccared Algorithms for 10000 nodes Benchmark Network

2.5.2 Evaluation of Similarity-CNM Algorithm with Real-World Networks

For this experiment, we used two real-world social networks, to compare our results against the original CNM algorithm results. The first social network is Flickr, which is a photo sharing social network. In Flickr, users can share and embed photographs in their own blogs. We used the data set provided by Cha, et al. [85]. The Flickr network consists of 2,570,535 nodes and 33,140,018 links between the nodes. We selected 10,000 nodes, to run our experiments. The second social network is the American Football Network which is a network of American football games between colleges (Division IA) during Fall 2000 season [86]. This network contains 115 nodes.

We focused on finding the value of Q_{max} since it represents the best partition in the network. We compare Q_{max} results obtained using the original CNM algorithm against the Q_{max} results we got from applying Similarity-CNM algorithm. We also compared the number of steps to reach Q_{max} for both algorithms. We noticed a difference in modularity as well as the number of steps used for both experiments. Similarity-CNM produces higher Q_{max} in fewer steps compared to the original CNM algorithm. Table 2.9 and Figure 2.12 show the values of Q_{max} for both Flickr and the American football social networks.

	Flickr	American Football
Original CNM	0.247936	0.577
Similarity-CNM	0.9253	0.813

Table 2.9: Comparing Maximum Modularity using CNM Algorithm and Similarity-CNM Algorithm for Real-World Networks

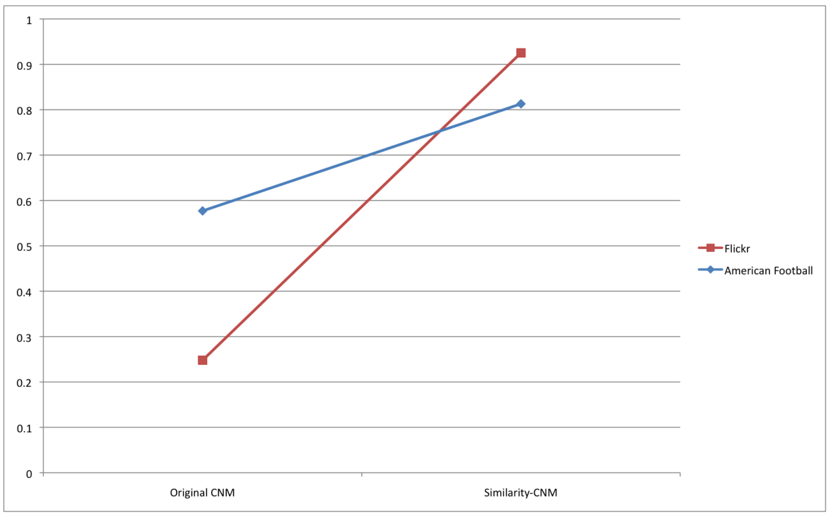


Figure 2.12: Maximum Modularity using CNM Algorithm and Similarity-CNM Algorithm for Flickr and American Football Networks

	Flickr	American Football
Original CNM	9879	108
Similarity-CNM	8843	100

Table 2.10: Comparing Number of Steps to Reach Q_{max} using CNM Algorithm and Similarity-CNM Algorithm for Real-World Networks

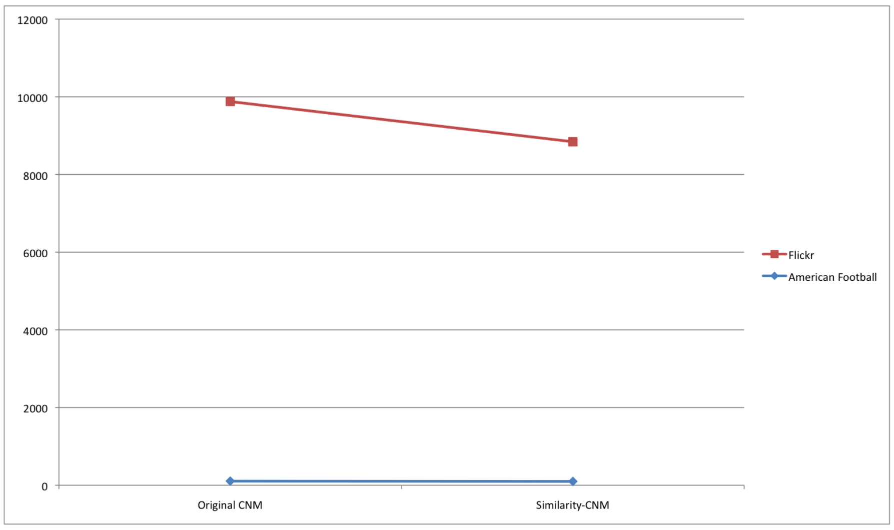


Figure 2.13: Number of Steps to Reach Q_{max} using CNM Algorithm and Similarity-CNM Algorithm for Flickr and American Football Networks

For the Flickr network, the maximum modularity value for Similarity-CNM outperforms the original CNM algorithm by more than 67%. While for American Football network, the maximum modularity value for Similarity-CNM exceeds by more than 23% the value obtained in the original CNM algorithm. Table 2.10 and Figure 2.13 also show that number of steps to reach Q_{max} is lower in Similarity-CNM than the original CNM algorithm. For Flickr, the number of steps when running the original CNM is 9879 steps, while in Similarity-CNM it is reduced to 8843 steps. For American Football, the number of steps is also reduced from 108 steps for original CNM to 100 steps for a Similarity-CNM algorithm.

We also applied the Similarity-CNM to different network sizes on the Flickr network, because it has high number of nodes, compared to the American football network, and thus we could observe performance scales for varying network sizes. Starting with 500 nodes and increasing the network size to 10,000 nodes, we compiled a range of performance results in Tables 2.11 and 2.12, and illustrated the corresponding graphic presentations of the results in Figures 2.14 and 2.15.

Although the Similarity-CNM requires consistently fewer steps than the original CNM, the scale of the steps increases faster in the Similarity-CNM. This contributes to the pre-processing that Similarity-CNM needs to perform on the nodes before applying the original CNM algorithm. A new virtual social network (similarity social network) will be generated, with fewer nodes and edges than the original one, each time we add new nodes to the network. This will decrease the steps to reach the maximum modularity (Q_{max}) value and the accumulated Q will reach maximum value in larger scales. Compared to the original CNM, the Similarity-CNM algorithm performs better and is more sensitive to network size.

Flickr (Nodes)	500	1000	2000	4000	6000	8000	10000
Original CNM	0.314201	0.196513	0.154273	0.200447	0.194078	0.194702	0.247936
Similarity-CNM	0.733643	0.900258	0.956798	0.976981	0.982534	0.983595	0.9253

Table 2.11: Comparing Maximum Modularity for different Flicker Network Sizes using CNM Algorithm and Similarity-CNM Algorithm

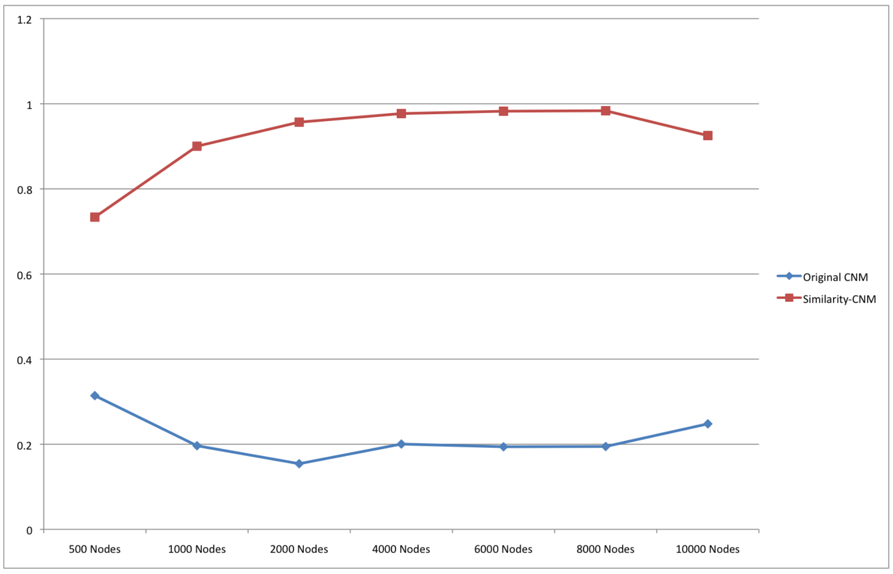


Figure 2.14: Maximum Modularity for Different Network Sizes using CNM Algorithm and Similarity-CNM Algorithm

Flickr (Nodes)	500	1000	2000	4000	6000	8000	10000
Original CNM	494	979	1973	3954	5929	7908	9879
Similarity-CNM	384	821	1717	3473	5257	7015	8843

Table 2.12: Comparing Number of Steps to Reach Q_{max} for Different Flicker Network Sizes using CNM Algorithm and Similarity-CNM Algorithm

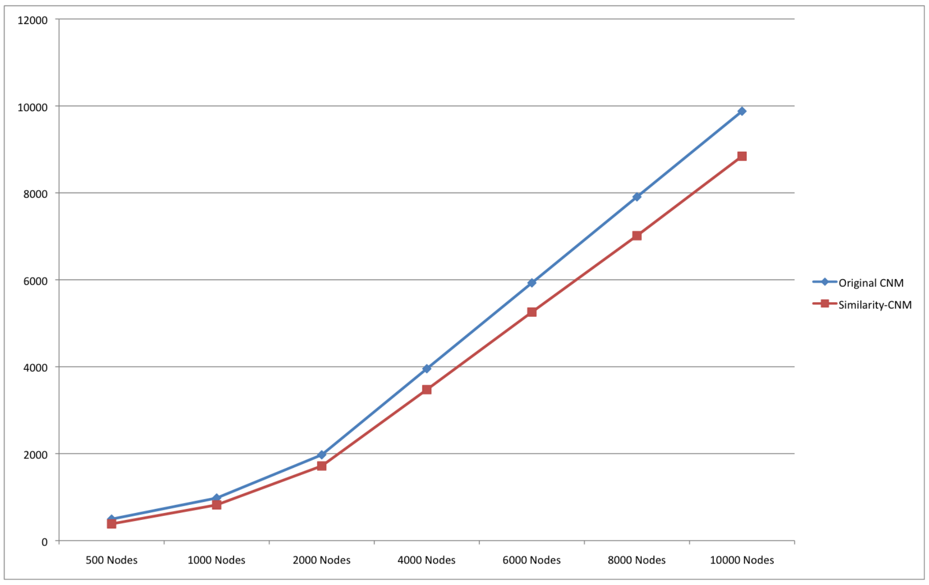
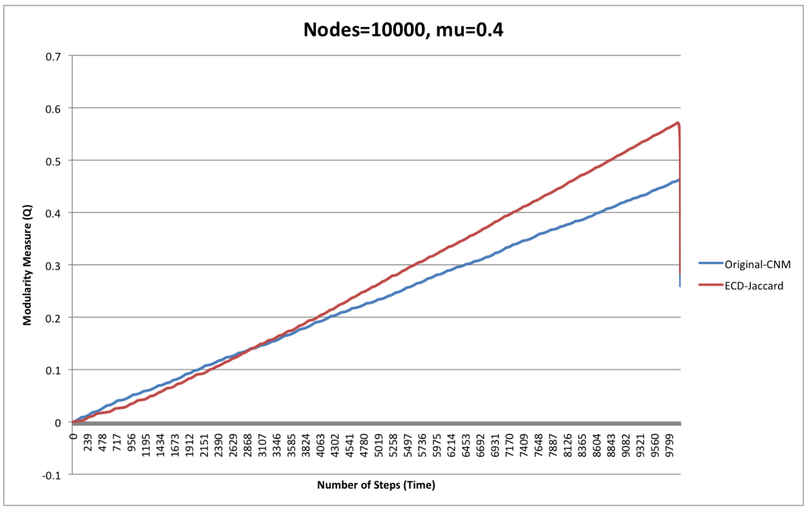


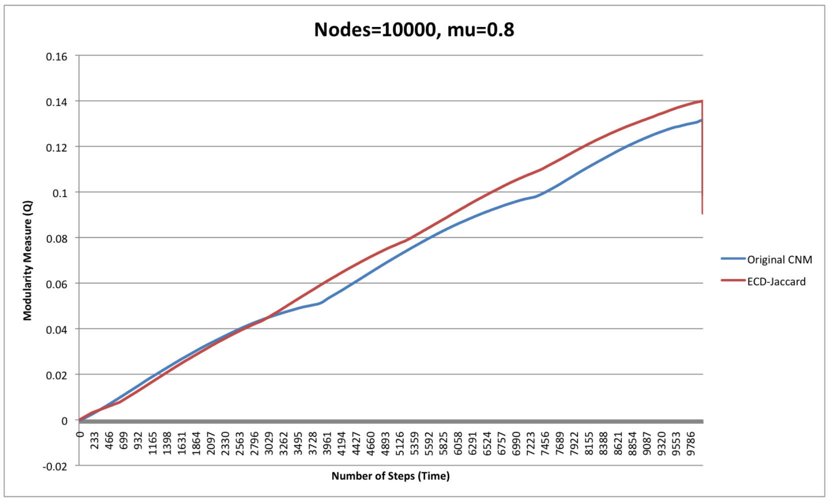
Figure 2.15: Number of Steps for Different Network Sizes using CNM Algorithm and Similarity-CNM Algorithm

2.5.3 ECD-Jaccard Algorithm Simulation with Artificial Networks

In this experiment, benchmark networks were used that have the same configuration as in Section 2.5.1. After generating six unweighted benchmark networks, we applied the Jaccard similarity technique and derived the edge weights for each network, in order to generate weighted networks. The CNM algorithm was applied on both unweighted and weighted networks. Figure 2.16 shows the evolution of the modularity (Q) over time for networks with μ value 0.4 and 0.8 respectively. We noticed that Q in general starts small then increases over time to reach its maximum point. Beyond this point, the Q value drops sharply as the randomly generated edges exceed the actual ones. By calculating the nodes similarity and assigning weights to the network edges we noticed higher Q values and better maximum Qs in weighted networks than in unweighted networks. The values of the maximum Q in weighted artificial networks outperforms the maximum Q values in unweighted networks by about 11% when μ value is equal to 0.4. Higher values resulted from feeding the CNM algorithm with the edge weights generated earlier. Adding similarity features to the network edges helps in grouping nodes in robust sets which ensures a better community structure.



(a) Maximum Modularity When $\mu = 0.4$



(b) Maximum Modularity When $\mu = 0.8$

Figure 2.16: Modularity (Q) Evolution Over Time for Different Mixing Parameters μ

By comparing the maximum modularity (Q) for all generated networks, both weighted and unweighted, we noticed that for networks with high mixing parameters (μ), the modularity is low due to weak community structures and less internal connectivity between the nodes. As the mixing parameter decreases, the modularity measure (Q) increases, resulting in a higher node density within communities. These results are depicted in Figure 2.17.

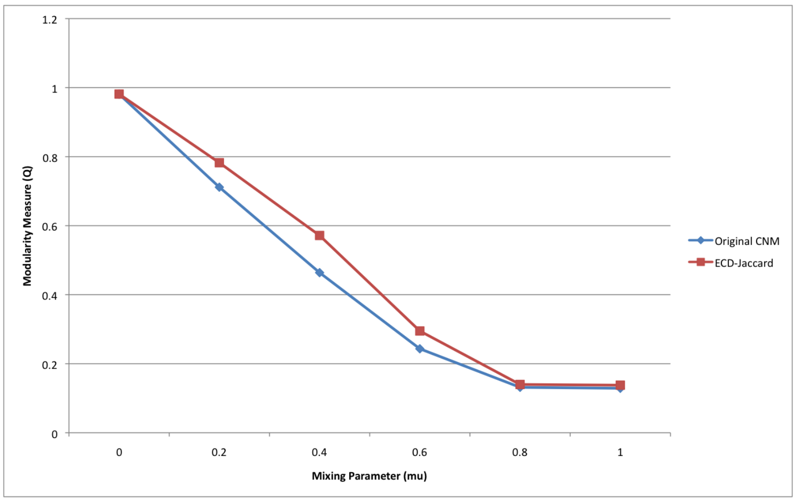


Figure 2.17: Maximum Modularity (Qs) for Different Values of µ

The normalized mutual information for the ECD Jaccard is 100%, 54% and 0.3% when $\mu = 0{,}0.4$ and 0.8, respectively as shown in Figure 2.11. When $\mu = 0.4$, ECD-Jaccard algorithm is able to detect 49 communities compared to 7 communities detected by the CNM (see Table 2.8).

2.5.4 Application of ECD-Jaccard Algorithm to Real-World Networks

We also applied these techniques to data collected from the real-world social network Flickr [85]. The data set was represented by an unweighted directed graph in order to match the previous experiment (i.e. artificial settings). From the data set provided [85], we selected 10,000 nodes and extracted the list of links (or edges) between these nodes. The nodes represent Flickr users. We generated the weighted version by applying the Jaccard similarity technique. Figures 2.18 and 2.19 show that adding similarity between the nodes as a weight to the edges also increases the modularity in real-world networks. The weighted network results in higher modularity (Q) which increased by almost 3% compared to the original CNM algorithm when the number of nodes is 10000. This confirms the range of performance results obtained earlier in the artificial network experiments.

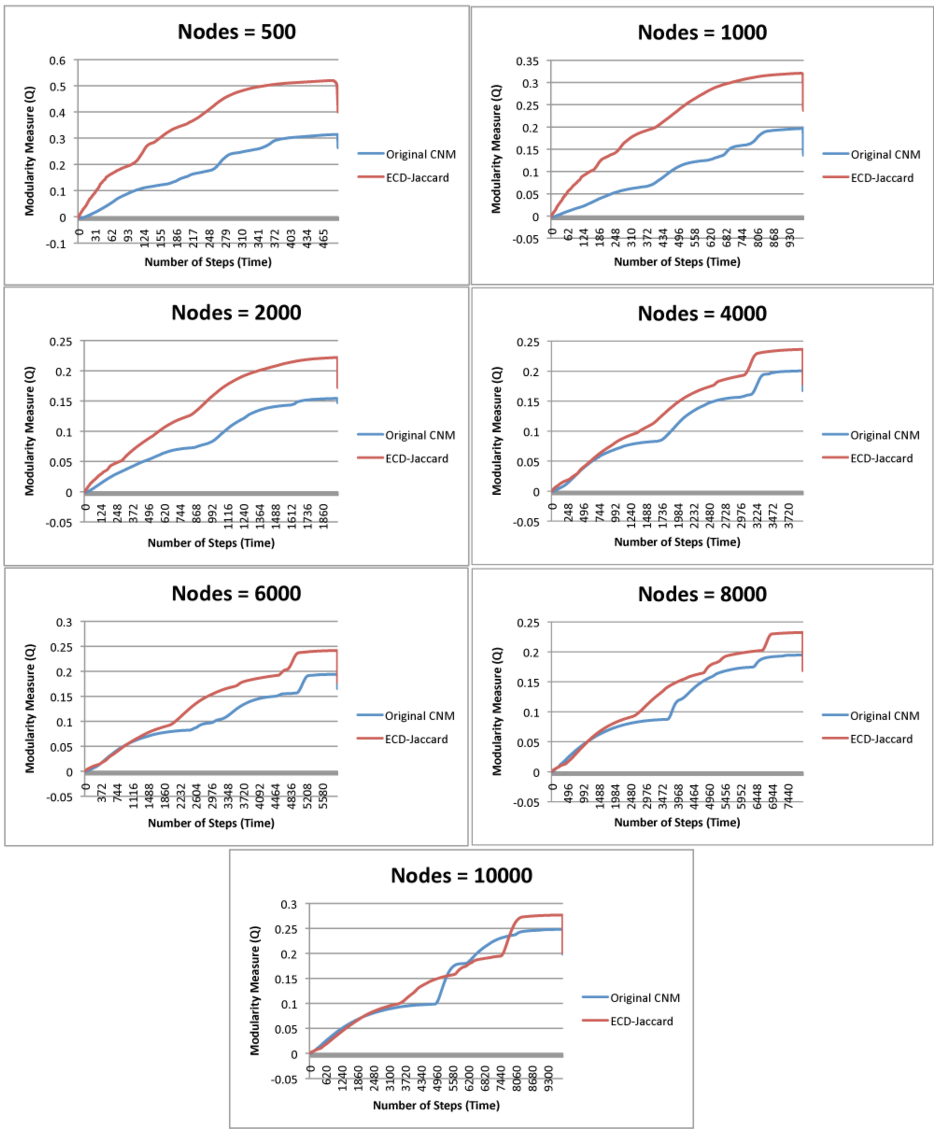


Figure 2.18: Modularity (Q) Evolution Over Time for Flickr's Real-World Network

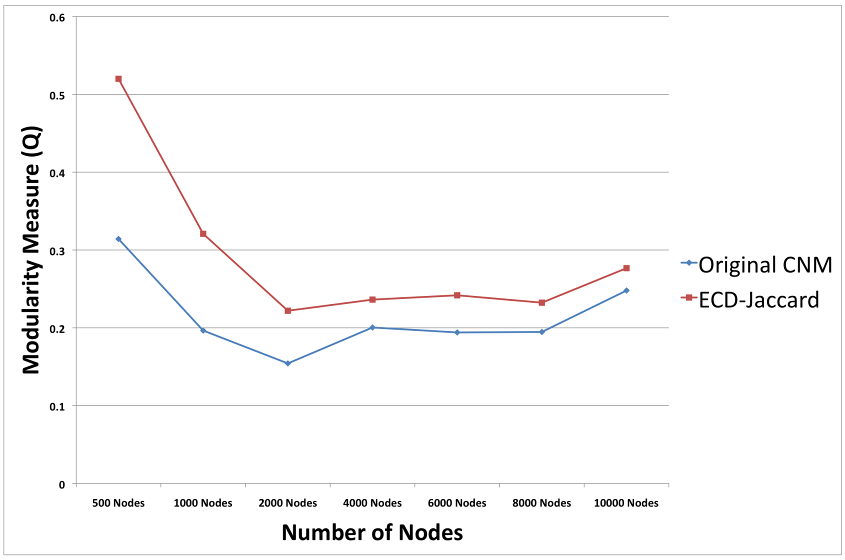


Figure 2.19: Modularity (Q) Evolution Over Number of Nodes for Flickr's Real-World Network

2.6 Related Works

Many algorithms for community detection have been proposed [55], and extensive comparative studies between these algorithms have been investigated previously [87, 53, 88, 89, 90, 91]. Newman and Girvan proposed a method for discovering communities based on hierarchical divisive algorithms [79], where an edge is removed iteratively from the network to split it into communities. However, these divisive algorithms were rarely used to study community detection at that time as most research was based on agglomerative algorithms due to various complexity issues. The main idea of Newman and Girvan's algorithm is to remove the edge with the highest betweenness. One speedy method to measure edge betweenness is the shortest-path betweenness by measuring all the shortest paths passing through a given link. Once an edge is removed, a recalculation of the edge betweenness is needed for all edges, which leads to substantial computational costs. The process continues until the desired communities with a target threshold are reached.

Fortunato, et al. [92] implemented a hierarchical clustering algorithm based on the work of Newman and Girvan [79]. They used the centrality measure to iteratively find and remove edges in the network. Their work shows that even though the algorithm runs in $O(n^4)$, it is powerful when dealing with mixed and hard to detect communities.

Newman [80] also proposed another qualitative method for identifying communities, called modularity. The modularity method uses a quantity function Q to measure whether a specific division is meaningful, when identifying communities. Their algorithm was simple with a reasonable running time (compared to other existing algorithms at that time) with the worst complexity of $O(n^2)$ on a sparse network of n nodes. This method has become very popular and is widely used. However, it has a resolution limit. Fortunato and Barthélemy [93] showed that modularity has an upper scale, which depends on the size of the network, and any module that is smaller than the scale might not be determined. Zhenping Li, et al. [94] proposed a quantitative modularity density measure to get over the limits of the regular modularity method proposed by Newman and Girvan. They used both the nodes and the edges in their method and showed that the optimization of the quantitative modularity density measure will not affect the network division process, resulting in better detection of communities. But this method is still NP-hard. Another solution for the resolution limit [95], suggests using a modified (multiresolution) version of modularity. Recently Lancichinetti and Fortunato [96] showed that multiresolution modularity also has limits related to merging small clusters when the resolution is low and splitting large clusters when the resolution is high.

Clauset, Newman and Moore [48] proposed another algorithm called CNM. The CNM algorithm is a bottom-up agglomerative clustering method that uses greedy techniques to combine and merge clusters in a network. The algorithm is similar to [80] and also gives similar results for detected communities. But it is more efficient in identifying communities since its time performance in worst case scenarios drops to $O(nlog^2n)$.

To the best of our knowledge, the use of weighting schemes to enhance community detection has rarely been explored. Khadivi, et al. [97] proposed pre- and post-processing steps to improve the Newman and Girvan algorithm known as

Newman Fast. They calculated the weights for each edge e_{ij} connecting any two vertices i and j in the graph. The weight represents the normalized product of the edge betweenness and the common neighbor ratio. After calculating the weights, the Newman Fast Algorithm uses a greedy method to maximize function Q. The Newman Fast Algorithm starts by representing each vertex as a community. Then communities are merged to maximize the value of Q. The algorithm ends when Q cannot be improved any more or all nodes have formed one single community.

An enhanced approach of the Newman Fast Algorithm was proposed by the same authors [98]. They proposed two parameters α and β that control the values generated by the edge betweenness and the common neighbor ratio. Different pairs of α and β will generate different results. Our approach is different than the methods proposed by [98], as we use a Jaccard Measure to compute similarities between nodes instead of using only common neighbors.

Berry, et al. [99] focused on studying the resolution limits discovered by Fortunato and Barthélemy [93]. However, they investigated this on weighted networks. Fortunato and Barthélemy found that there was a limit for communities that can be detected, represented by $\sqrt{|E|/2}$, where $|E|$ is the number of edges in the network. The authors stated the resolution limit for weighted networks to be $\sqrt{W\varepsilon/2}$, where W is the sum of the weights in the network and ε is the maximum weight of an edge connecting two different nodes from two different communities. They proposed a modified version of the CNM algorithm called wCNM. They calculated the edges' weight based on the number of iterations of length k. Our approach is different from Berry, et al., as we generate a virtual social network based on nodes similarity and then use it to detect the final communities through a CNM algorithm.

Yan and Gregory [100] also studied the effect of adding weights to the network results generated by community detection algorithms. They used the common neighbor technique to calculate the edges' weights. The common neighbors measure is based on finding the number of common neighbors for each two vertices in a network.

Another approach was proposed by De Meo, et al., in [101]. They proposed a k-path centrality technique that is computed using random paths of most k lengths. This approach was not designed for any specific community detection algorithm [97, 98]. Instead, it works with any algorithm that deals with weighted networks. Our approach is different as we actually propose two methods to enhance community detection. In the first proposal, we generate a virtual social network based on nodes similarity as introduced earlier. In the second enhanced approach, we use the Jaccard Coefficient to calculate similarity between nodes. In fact both proposed approaches are based on the Jaccard Measure. The first proposal considers the edges instance between the nodes. For example if there is a link between nodes a and b then the adjacency matrix value is 1, otherwise it is zero. In general, the Jaccard and the common neighbor techniques look similar, but in fact there is a major difference between them. The Jaccard measure computes the similarity between every two nodes, by dividing the intersection of the common neighbors by the sum of all neighbors of the two nodes. Whereas the common neighbor technique takes only inter-connections into consideration [102].

2.7 Summary

In this chapter, we discussed clustering techniques in social networks since these techniques are the basis for detecting communities. Two complementary approaches were suggested for community detection: Similarity-CNM and ECD-Jaccard, to provide enhanced community structures. Both algorithms use an offline pre-processing step before applying the CNM algorithm. We used a benchmark network generator as developed by Lancichinetti, et al. [84]. We also used a real-world social network, Flickr, to compare our results to the original CNM algorithm results. For both algorithms, Similarity-CNM and ECD-Jaccard, we observed good results compared to the results generated from a CNM algorithm. We found that even though our algorithms run in $O(n^2)$ time, this offline pre-processing manipulation can detect better communities.

Chapter 3

Social Influence

3.1 Introduction

Online Social Networks (OSNs) have grown in popularity since they were introduced a decade ago. Millions of people participate and register in online social networks such as Facebook, LinkedIn, Flickr, MySpace, and Twitter. Facebook itself accounted for more than 1 billion active user in 2012[1]. These social networks have a great impact on people's lives at different levels, and in a variety of ways. One use of OSNs is in reporting adversity and boosting awareness about situations, especially in places that lack physical communication facilities, due for example to nature disasters such as hurricanes and earthquakes. People are increasingly using social networks to spread information during crises because these networks are handy and easy to use. Acar and Muraki [103] studied posts on Twitter, (called tweets), two weeks after the devastating Tohoku earthquake that resulted in an overwhelmingly destructive tsunami in Japan during March 2011. They found that people in the affected areas had a tendency to post tweets related to their situation, while people in remote area post tweets to let their followers know they are safe. Another widespread use of OSNs occurred during political protests in Tunisia and Egypt in January 2011, where massive anti-governmental demonstrations forced dictatorships to fall. What is interesting in these events is that social network bloggers did

[1]https://newsroom.fb.com/news/2012/10/one-billion-people-on-facebook/

not use OSNs to advertise their webpages or encourage people to write about their frustrations, but to engage people and motivate them into taking action not only online but in real world too, which illustrates the influential power and impact of social networking.

In real life contexts, an influencer is a person who is followed by many people and has the power to make changes in a community. The same aspects occur in OSNs contexts, as they form a large social space where people are engaged together to build relationships and expand their connections with others. These OSNs have the same traits of real-life communications and many people thrive socially in OSNs as they do in their real life. In previous years, influential people were those who had many friends. This idea evolved as influencers started not only to have many friends, but also to actively engage their friendship community in action. Currently, many influencers drive discussion topics about a specific topic or brand [104]. This kind of influence was the main building block of the interest graph: a network of people who are interested in each other's content [105]. Interest-graphs help with branding products and services by targeting influential people in social networks [105].

In this chapter, we address social influence by evaluating different models used to measure influence probability. Conceptually, OSNs are related to graph theory [36], computer science and the social sciences [106]. To study and analyze these networks properly, a combination of these disciplines needs to be considered. Social networks can be modeled as a graph that contains nodes representing members and edges corresponding to the relationship types between the nodes (e.g. friendship). Social Networks Analysis (SNA) [107] can help in addressing the sources and distribution of influential power in social networks, based on the structure of the network. The influential power of a user rises with his relationship to other influential users in the network. Sociologists have studied the power of a specific node in a network by addressing the attributes of centrality using SNA. They look at degree, closeness and betweenness centralities [107]. Nodes with high degrees, high closeness and high betweenness will have greater influence. Figure 3.1 shows a sample social network graph and the edges between the nodes. The graph shows clusters and central nodes that can be sources of great influential power when evaluating their social influence. One drawback of measuring influence based on SNA is that centrality is based on the structure of the network, while influence should be based on the dynamics and changes that occur in the OSNs connections and links.

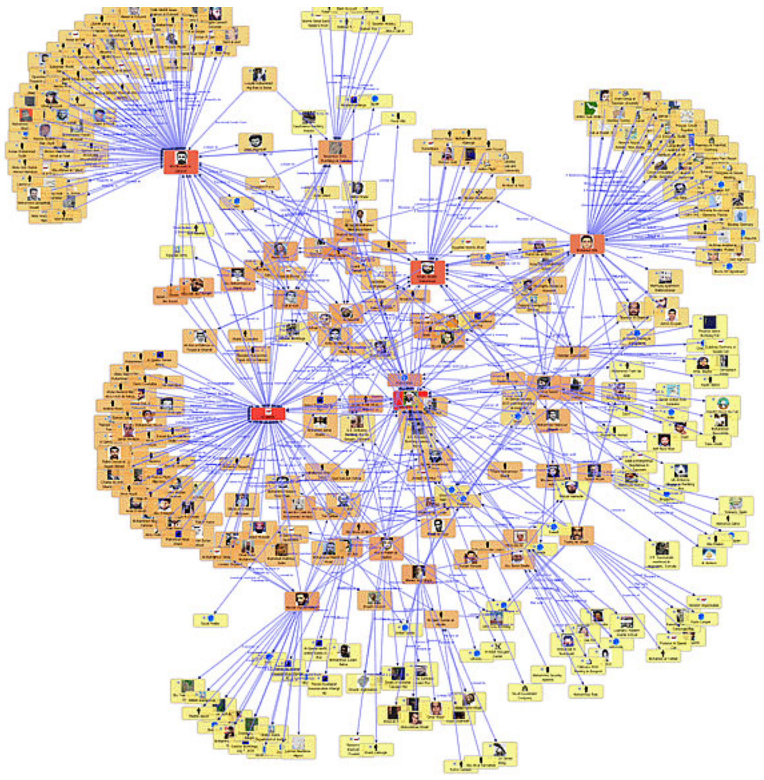


Figure 3.1: Social Network Diagram

A better understanding of the evolution of social networks leads to a better understanding of the community structure and social influence [108] of these networks. This investigation helps in conducting different activities around OSNs based communities such as targeted advertisements, and item recommendation to OSNs users. Research works out in this area is sparse, and spans multiple disciplines. As well as our efforts to summarize the state of the art surrounding social influence in OSNs, we also evaluate different approaches and classify them in order to understand commonalities and distinguish differences.

The rest of this chapter is organized as follows: Section 3.2 provides basic information about social influence through defining influence in OSNs contexts. Section 3.3 discusses suggested social influence models as well as their limitations, strengths and challenges. Section 3.4 provides a survey of social influence related works on OSNs and state the problem of measuring influence probability. Section 3.5 provides a classification of influence propagation algorithms. In Section 3.6 we compare two benchmark influence propagation algorithms, Independent Cascade and Leaner Threshold. Finally, Section 3.7 concludes the chapter with some possible future extensions to the social influence models presented.

3.2 Influence in Online Social Networks

Social Influence has been studied by sociologists and social psychologists since the early years of the 20th century [109]. It started in 1898, with the first experiment by Norman Triplett on the phenomenon of social facilitation [110]. This theory implies that people tend to do well in the things they are good at when they are watched by others [110]. One of the main theories of social influence was proposed in 1950 by Leon Festinger, it is called Cognitive Dissonance Theory. This theory is related to how thinking can affect our behavior [111]. In 1959, French and Raven discussed social power and provided a formalization for the social influence concept [112]. Research was more mature in both theory and method during the 1980's and 1990's [109].

Social Influence has been studied in different disciplines and has historical roots in sociology, through studying opinion formation and the diffusion of innovations [27, 28]; in economics, where social influence is represented by theoretical models that show how individuals are inclined to coordinate their economic decisions [29, 30]. Recently, digital social influence research has started to attract more attention due to the availability of many important applications. For example, computer scientists have developed models of social influence to support applications such as viral marketing [31, 3, 32], the spread of online news [33, 34], and the growth of online communities [35].

3.2.1 Social Influence Definition

Sociologists define social influence as a “change in an individual’s thoughts, feelings, attitudes, or behaviors that results from interaction with another individual or a group” [112]. Social influence occurs when an individual changes his/her behavior after interacting with other individuals who tend to be similar or superior.

Social influence develops social correlation, which is divided into three categories [113]:

- **Influence**: where the user performs an action based on his/her friends’ recent actions. For example when the user purchases a product because one of his friends just bought or recommended that product.
- **Homophily**: A user chooses friends who share the same characteristics [114, 115], which leads to performing the same actions. For example two people who have Xbox are more likely to be friends.
- **Confounding factors**: or external influence that affects individuals who are located near each other in the social network. One example would be when two users live in the same city, which makes them perform the same activities like taking the same photos and posting them with the same tags in an online photo social network such as Flickr.

Social influence across OSNs can help in spreading different behaviors, ideas, and new technologies throughout the network. For example, a fashion company might provide coupons to the most influential users in their social network or take advantage of these users to promote a new product. Different researches has been conducted to study the methods of leveraging social influence [3] and the effect of influence on product growth [116]. This leads to the process of carefully choosing targets with high influential power as a good marketing strategy that could lead to high acceptance levels for a certain product among users of the social network. Social influence is becoming a complex and subtle force that governs the dynamics of all social networks. Therefore, there is a need for methods and techniques to analyze and quantify social influence.

The rich properties and components of social networks paved the way for better analysis of individual user actions, that leads to further profiling users' behavior in OSNs. Through their behaviors, people can influence other users to do specific actions. This is a powerful process that can generate revenue or incite global actions. Social influence appears as a social correlation pattern where the actions of a user can urge his or her friends to behave in the same way [117]. OSNs provide different properties that make it possible to study user actions that influence community behaviors. The availability of rich interactions between users and the data that results from such interactions facilitate social influence analysis. In Section 1.2 we described social network structures and introduced some properties that help in a better understanding of social influence.

3.2.2 Basic Measurements for Influence Strength and Power

Social networks are modeled as graphs $G = (V, E)$, where V is the set of nodes in the network and E is the set of edges. The nodes are related to the users and the edges represent the relationships between these users in the network. Influence strength can be related to a node or an edge in the network. For example some nodes in the network might have a higher influence than other nodes, let's say that node A has a high influence and higher edge strength on node B. This strong influence will make node B behave similarly to node A. In this section we will present the basic measures of strength for edge and node levels.

Edge Strength

The edge or tie strength concept was introduced by Granovetter [118]. For edge level there are two different types of ties, strong ties and weak ties. The tie strength depends on the number of overlapping friends or neighbors between the two nodes [118]. The larger the overlap the stronger the ties between the nodes. The strength between two nodes A and B can be defined in terms of a Jaccard coefficient as follows [119]:

$$S(A,B) = \frac{|\, n_A \cap n_B \,|}{|\, n_A \cup n_B \,|}$$

where n_A and n_B are the neighbors of nodes A and B, respectively. There are other measurements to determine the tie strength such as embeddedness [120]. Strong ties represent a trust relationship between nodes. While weak ties occur between acquaintances when the overlap is small and restricted information is shared between the nodes such as private and personal details and posts.

Node Strength

The importance of the node in the network is measured through centrality. Nodes with high centrality have higher influence in the network than nodes with less centrality power. Here we distinguish three levels of centrality: degree, betweenness and closeness.

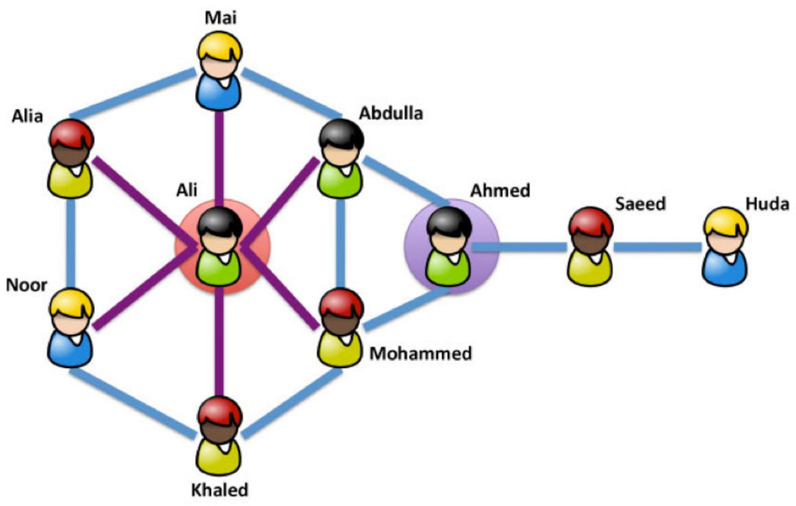


Figure 3.2: Degree Centrality

Degree Centrality

is the number of ties that a node has [107]. In Figure 3.2, node Ali has the highest degree centrality, because it is the node with the highest number of ties or edges.

This means he is quite active in the network. However, he is not necessarily the most influential person because he is only directly connected within one degree to people in his group. He has to go through Ahmed to get to other connections.

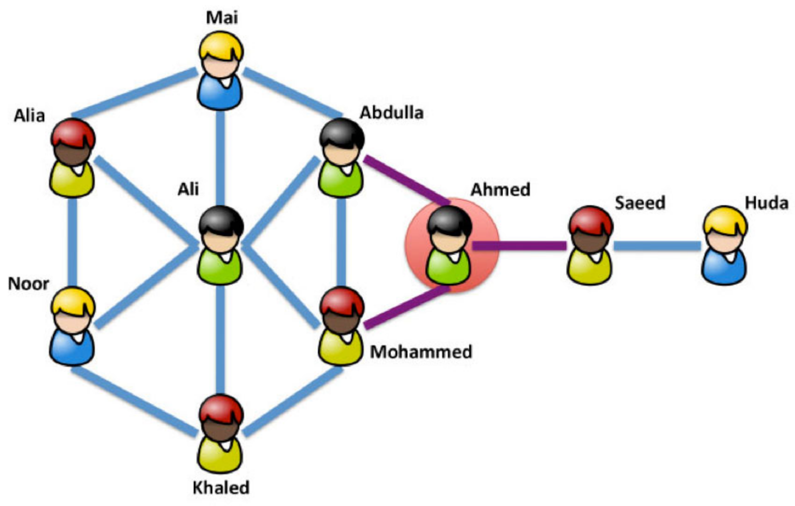


Figure 3.3: Betweenness Centrality

Betweenness Centrality

occurs when a node falls in a favored position between two groups in the network [107]. In Figure 3.3, Ahmed has the highest betweenness because he is between Abdulla, Mohammed and Saeed, who are between other nodes. Abdulla, Mohammed and Saeed have lower betweenness because they are essentially within their own groups. So Ahmed has potentially more influence in the network. Betweenness represents a single point of failure; when the node with highest betweenness centrality is removed from the network the ties between groups separate.

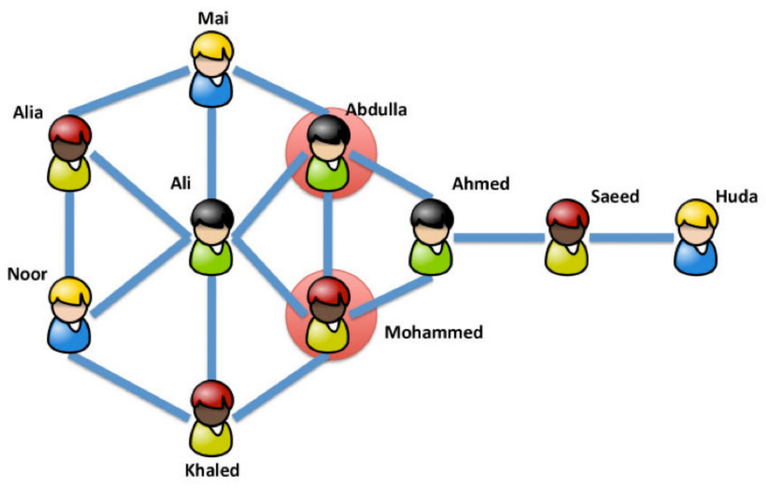


Figure 3.4: Closeness Centrality

Closeness Centrality

measures how quickly a node can access more nodes in a network [107]. In Figure 3.4, Abdulla and Mohammed have the highest closeness centrality because they can reach more entities through shorter paths.

3.2.3 Social Influence Analysis

There are different considerations for modeling influence in social networks. Edge and node strength are typical attributes used to analyze influence in social networks. In addition, the following are additional analytical considerations:

- ***Multi-topics:*** Social influence will have different effects on different topics discussed in the social network. For example, assume two neighbors A, specialized in data mining and B, specialized in programming. A will have high influence on B when the topic is related to data mining while B will have higher influence on A when the topic is related to programming.

- ***User actions:*** considering user actions and past behaviors while measuring influence.

- ***Scalability:*** The numbers of nodes in OSNs increases rapidly. Therefore there is a need to develop methods that scale well with large datasets [113].

As far as we know, limited research has compared modeling techniques to specify their limitations and challenges. In this chapter, we will address these issues based on social influence models. We also propose a categorization of these models based on general criteria to compile their common features and distinguish their differences in a single snapshot.

3.3 Structures And Models of Social Influence

Some theoretical and empirical works have been conducted in order to understand users' behavior when correlated to their friends' attributes in social networks. Backstorm, et al. [35] observed the process of joining an online community and noticed a correlation between a user joining an online community and the number of friends who are in the community. Another study by Marlow, et al. [121] observed tag usage in Flickr. They noticed a correlation between the tags assigned by a user and those assigned by his friends in his social network. These provides evidence of influence between users' and their friends.

The spread of influence can be modeled through probabilistic frameworks [2]. While a behavior is spreading through the social network we need to estimate the probability that a particular individual will embrace the new behavior, given that k of his/her neighbors in the social network have done so. Neighbors refer to people who have a direct edge or tie between them in the OSNs. At any point in time t, users would be 'adopters' or 'non-adopters' of the behavior based on whether they adopt the new behavior [2].

The properties of social networks enable probability evaluation of user behaviors especially when those behaviors are spread over a large population. For example the probability of a person purchasing a product given that k of his or her friends recommended that product [32]. Another example would be the probability of joining an online community as a function of the number k of neighbors belonging to community [121, 122].

If we have a social network with the intention of influencing the individual users, as we want to introduce a new product, then a viral marketing strategy could start by targeting the most influential users in the network. This will generate a chain-reaction of influence driven advertisement campaigns. By using this method, reaching a very large proportion of the network can occur with very small marketing costs.

3.3.1 Social Influence Structure

The problem of influence maximization can be expressed as follows: "given a network with influence estimates, how to select an initial set of k users such that they eventually influence the largest number of users in the social network" [123].

This influence problem can be formally stated as follows: given a social graph that is undirected $G = (V, E, T)$ where V represent the set of users in the network, E is the set of edges in the network and T is the matrix of timestamps at which the social ties were created, matrix social ties represent the links and relationships between the nodes in the social network. A tie between users u and v is represented by an undirected edge $(u, v) \in E$. Each edge is labeled with a timestamp at which the edge was created. Assuming that social ties are never broken [123], the labeling function can be represented by $T : E \rightarrow N$.

A log of actions, is maintained where an action could be joining an online community or purchasing a product. This is formulated as *Actions(User, Action, Time)* where a tuple (u, a, t_u) indicates that user u has performed action a at a time t_u. The log contains all the actions performed by all users in V of the social graph G. Let A represents the actions set, A_u represents the number of actions performed by user u, and $A_{u\&v}$ is the number of actions performed by both users u and v and $A_{u|v}$ represents the number of actions that either u or v performed. This can be shown through the following formula $A_{u|v} = A_u + A_v - A_{u\&v}$. We also use A_{u2v} to denote the number of actions propagated from u to v [123].

Definition 1 formally introduces the action undertaken between users in graph G.

Definition 1 (Action Propagation). We say that an action $a \in A$ propagates from user u to v if: (i) $(u, v) \in E$; (ii) $\exists (u, a, t_i), (v, a, t_j) \in Actions(V, A, T)$ with t_i < t_j; (iii) $T(u, v) \leq t_i$. When this happens we state the predicate $prop(a, u, v, \triangle t)$ where $\triangle t = t_j - t_i$.

Definition 2 shows the propagation graph [123] of each action. This leads to a natural notion of a propagation graph, defined next.

Definition 2 (Propagation Graph). For each action a, we define a propagation graph $PG(a) = (V(a), E(a))$, as follows: $V(a) = \{v \mid \exists t\, (u, a, t) \in Actions(V, A, T)\}$; there is a directed edge $u \xrightarrow{\triangle t} v$ in *E(a)* whenever $prop(a, u, v, \triangle t)$.

The propagation graph of an action is a directed graph, which contains all the users who performed that action, with the edges connecting them according to the direction of propagation.

3.3.2 Social Influence Models

Although many models have been proposed to address the problem of measuring influence probability, there are limited models to contrast their strengths and limitations. Sun and Tang [119] introduced a survey of social influence analysis models and algorithms for measuring social influence. They discussed influence maximization and its application in viral marketing. They focused on the computational aspect of social influence analysis by calculating a selection of people who are similar to each other (for example two users who have the same opinion) and influence that leads users to adopt behaviors experienced by their neighbors (for example changing the opinion of a user to agree with one of his neighbors). They also provided methods to measure the weight of influence. Our survey approach categorizes social influence models into four categories: 1) static models, 2)dynamic models, 3) diffusion models and 4) models based on user behaviors. We also contrast different influence models stating their strength and limitations.

OSNs are still new and need to be fully analyzed. OSN models should represent and satisfy some inherent properties introduced in Section 1.2.2. As a result modeling social influence in OSNs is still in its infancy. There are no standard models for representing influence, which leads to difficulties in analyzing large-scale networks based on social influence. In this section, we will look at different models of social influence and compare the results of each model to determine the most accurate way to measure the probability $(p_{u,v})$ with which a node u is influenced by its neighbor v. We discuss the strengths and limitations as well as the different challenges of the models.

Generally there are two basic categories to represent influential models in social networks. static influence models are the simplest and easy to test. They assume the probability of influence is static and time-independent. Only the current state of the network and the most influential nodes at that state are considered. The second category of models is labelled as dynamic influence models and assumes that influence changes over time. We will see later that the models in this category are the most accurate as they can tell the history of a specific network and identify the most influential nodes for spreading information, but they are very expensive when tested on large data sets as they take long time to execute for large social networks.

Other categories of social network models discussed in this paper are categorized as linear threshold models and independent cascade models. Models based on greedy algorithms and past user behaviors such as topical affinity propagation models are also addressed in this paper. Figure 3.5 shows the Social Influence Models we will discuss in this section.

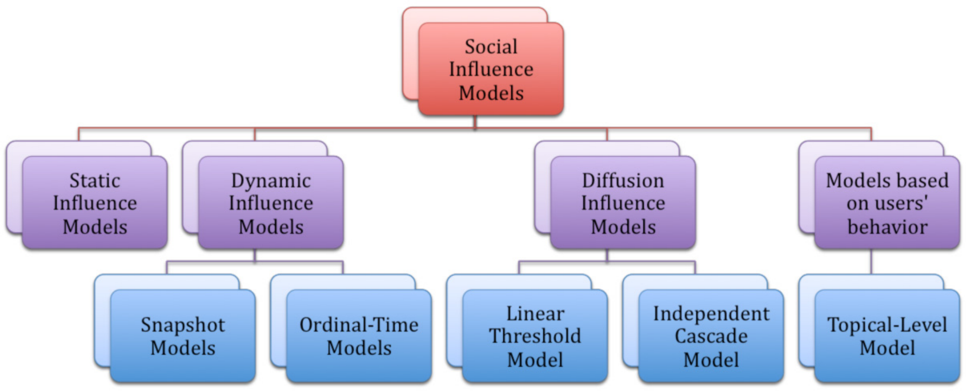


Figure 3.5: Social Influence Models

The challenge that any researcher faces is how to compare models of different categorization and state the relationship between them. Especially when the relationship between the models is ambiguous. Thus we aim to clarify this ambiguity by explaining and finding similarities in each.model.

Next, we will briefly introduce each category.

Static Influence Models

Static Influence Models are independent of time and used to capture the most influential nodes. Therefore the network size is fixed. One instance of this model is based on Bernoulli Distribution. The success state is labelled *n=1* and occur with probability p. In social influence a specific node u has a fixed probability to influence its inactive neighbor v. If it activates the neighbor then this is a successful attempt and otherwise failure. Each attempt can be shown as a Bernoulli trial. Figure 3.6 shows a sample illustration to explain Bernoulli trials. The influence probability can be estimated using a Maximum Likelihood Estimator (MLE) [123] as the ratio of successful attempts over the total number of trials:

$$p_{u,v} = \frac{A_{v2u}}{A_v}$$

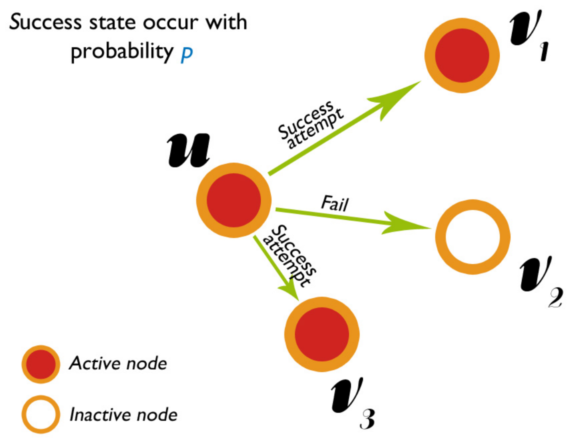


Figure 3.6: Bernoulli Distribution: Node u Will Have Fixed Probability to Influence its Inactive Neighbor v_1, v_2 and v_3. If u Attempt is Successful, Node v Will be Activated Otherwise Node v Will Remain Inactive

Dynamic Influence Models

In real-life, influence changes over time and may not stay static. For example, users' opinions could change over time. When a user is influenced by his/her neighbors to join a community, she/he is initially excited to join that community, but over time that user might be less excited. To represent dynamic influence models, we discuss two models of social influence. The first one is based on capturing a small set of 'snapshot' observations and the second one is based on detailed temporal dynamics. These two models can be represented as a function of the number k of neighbors who have adopted a new behavior [2]. The individual become $k-exposed$ to the behavior at specific time t if it is a non-adapter at time t but surrounded with k neighbors who are all adopters at time t.

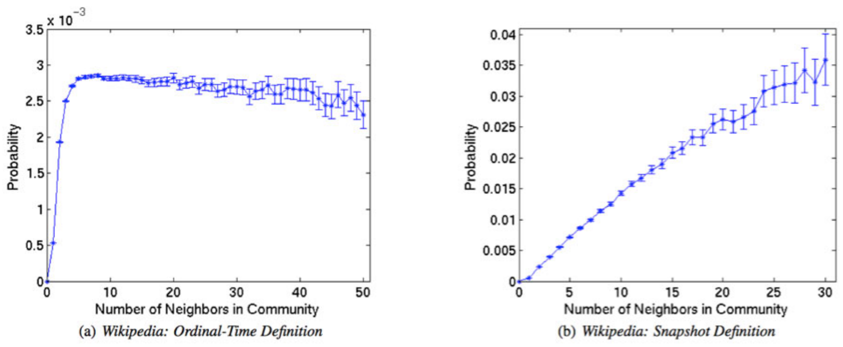


Figure 3.7: The Probability of Editing an Article in Wikipedia [2]

Snapshot Model

To represent this model we need to consider two snapshots of the social network at different points in time [2]. Consider then the set of all individuals who are $k - exposed$ in the first snapshot. Let $p_s(k)$ be the fraction of individuals in this set who have become adopters by the time of the second snapshot [2]. To further clarify, imagine that all $k - exposed$ nodes in the first snapshot will flip a coin $p_s(k)$ to decide wither to adopt the behavior or not. Based on different experiments on Wikipedia (a free, web-based, collaborative, multilingual encyclopedia), LiveJournal (a virtual community where Internet users can keep a blog, journal or diary) and engage in email correspondence [2, 35, 124] the snapshot curve (shown in Figure 3.7b) shows that the influence increases with more links, but the marginal influence of each additional link is slowly decreasing [2]. There are studies that used snapshot models to compute influence probabilities such as [124, 35, 122], though they used a large number of snapshots requiring substantial computational sources.

Ordinal-Time Model

To represent this model we need to consider a time sequence as it evolves over time. A new link is created in the network or a new individual adopts a new behavior. For each k, consider the set of all individuals who were ever $k - exposed$ at any time, and define $p_0(k)$ to be the fraction of this set that became adopters before acquiring a $(k+1)^{st}$ neighbor who is an adopter [2]. To clarify imagine that a non-adopter acquired the k^{th} neighbor who is an adopter, by flipping a coin $p_0(k)$, the non-adopter will decide to adopt or not. The curve of ordinal time in Figure 3.7a shows that the first five links have greater impact, but after some propagation the subsequent links impact stabilizes. This feature is similar to a power of low distribution. In both of the above, cases there is a need to determine the maximum-likehood values of these probabilities $p_0(k)$ and $p_s(k)$.

Comparing different models and their relationships reveals interesting performance thresholds and application domains. Generally the snapshot model is widely used as it captures an observation without the need to perform moment-by-moment measurements [2]. Although there is no apparent relationship between the snapshot and the ordinal-time models, the shape of ordinal time can be approximated from data in single snapshot. Experimental analysis show that accurate result occurred with more snapshots.

Diffusion Influence Models

These models are used when adopting behavior depends on knowing the number of neighbors who adopted the same behavior. In [31, 125] Domingos and Richardson proposed a framework for the propagation of influence when addressing the problem of identifying influential users. They proposed a probabilistic model of interaction and heuristics to select the influential users in the context of viral marketing, and confirmed their approach through an empirical study. Their idea was based on how to find the most influential individuals and target them to advertise a new innovation or a product. In a large cascade, they will influence their friends and friends of friends. Market customers are represented as nodes in social networks and customer influence is modeled as a Markov random field. These Diffusion models can be used to optimize marketing decisions. Kempe, et al. [3] revealed that the problem of selecting influential sets of individuals in most influence models is NP-Complete. This set of individuals should be chosen to generate the maximum influence during the influence diffusion process. Approximation algorithms are used to solve the problem of influence maximization. In some influence models the greedy algorithm will select the set of individuals with approximation $(1-1/e-\varepsilon)$ [3], where e is the base of the natural logarithm and ε is any positive real number. In their work Kempe, et al. focused on two influence diffusion models that are, the linear threshold model and the independent cascade model.

Linear Threshold Model

Granovetter and Schelling [126] were among the first to propose the threshold approach to capture influence. In a Linear Threshold Model a weight $b_{u,v}$ is used to measure the tendency of a node u to be influenced by each neighbor v such that $\sum_{v\,neighbour\,of\,u} b_{u,v} \leq 1$. Starting with an initial set of active nodes A_0, Then the influence propagation continues as follows: each node u is assigned a threshold θ_u randomly from the interval [0, 1]; the threshold represents the weight fraction of u's neighbors that must adopt the behavior (be active) in order for u to become active and adopt the same behavior. At timestamp t, all nodes that were active in time $t-1$ remain active, and we then activate any node u for which the total weight of its active neighbors is at least θ_u; where

$$\sum_{v\,active\,neighbour\,of\,u} b_{u,v} \geq \theta_u$$

The thresholds θ_u represents the tendency of nodes to adopt the new behavior when their neighbors do [3]. Figure 3.8a-b shows an example of the Linear Threshold Model Process.

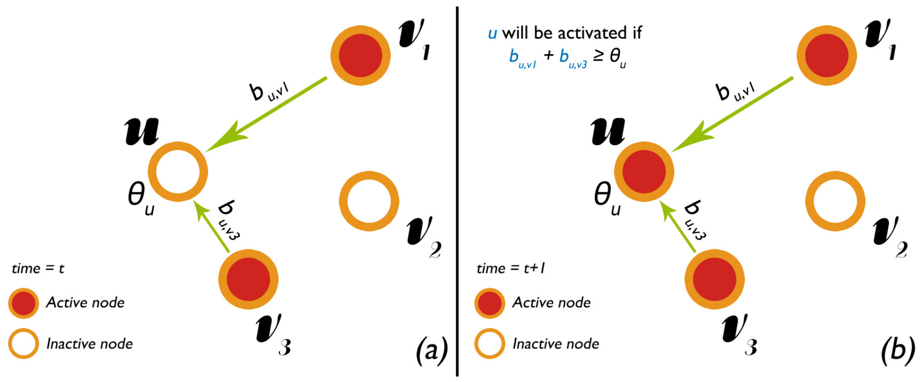


Figure 3.8: Example of Linear Threshold Model Process

In their experiment, Kempe, et al. [3] compared their greedy algorithm with node degrees and centrality within the network, as well as incorporating random nodes. Based on their experiment, their greedy algorithm outperforms the degree and distance centralities because these two features do not consider the dynamics of social networks and focus on the structure of the network to emphasize influence. Random nodes do not generate good results in a linear threshold model. Figure 3.9 shows the result of these experiments.

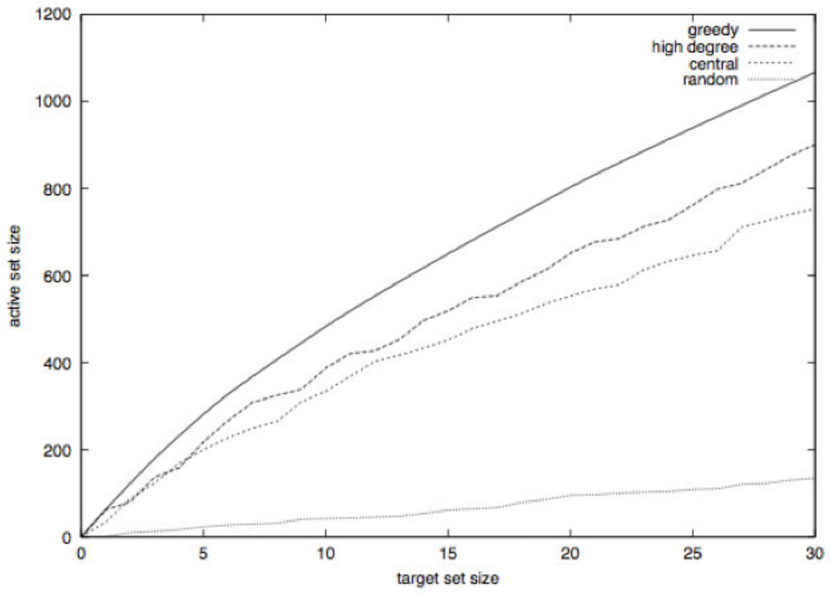


Figure 3.9: Results For The Linear Threshold Model [3]

Independent Cascade Model

An independent cascade model starts with an initial set of active nodes A_0. This set of individuals should be chosen to generate the maximum influence during the cascade diffusion process. The process occurs in discrete steps as follows: when node u becomes active for the first time in timestep t, it is provided with one chance to activate each of its currently inactive neighbor v; in that case u is called *contagious* which means it has the ability to affect other nodes as shown in Figure 3.10a. Node u succeeds in influencing its neighbor v with a probability $p_{u,v}$ independent of past history. If u succeeds, then v will become active in timestep $t+1$ as shown in Figure 3.10b; but whether or not u succeeds, it cannot make any further attempts to activate v in future rounds[3]. The same process continues until u's communicate with all neighbors to influence attempts and there are no more contagious nodes.

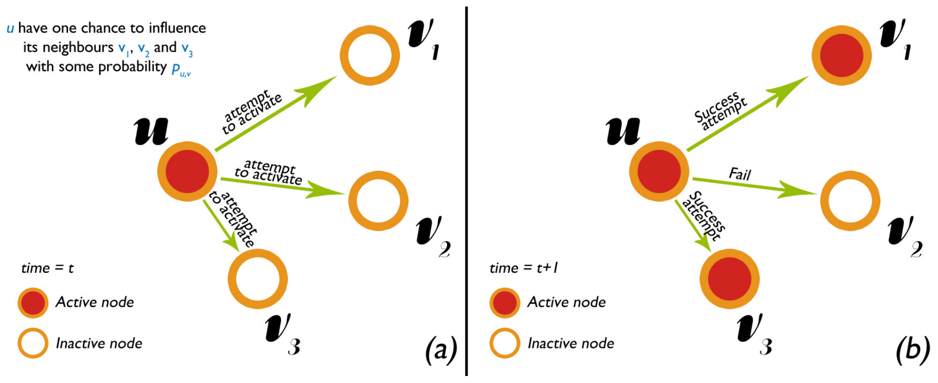


Figure 3.10: Example of Independent Cascade Model Process

Based on Kempe, et al.'s experiments on independent cascade model, we can see that the greedy algorithm still outperforms the degree and centrality within the network. Interestingly, random nodes performed well on independent cascade models, as shown in Figure 3.11.

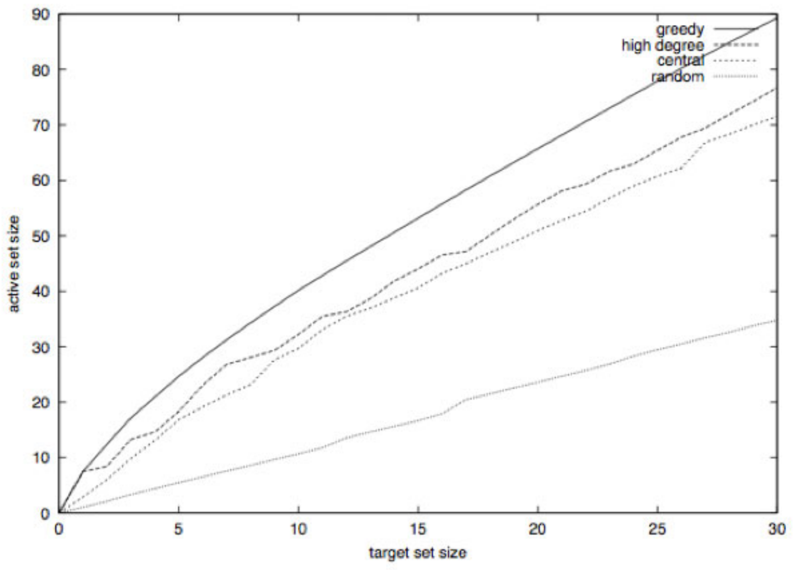


Figure 3.11: Results of Independent Cascade Model [3]

Models of influence based on users' behavior

The models discussed above are based on the influence model proposed in [3], where the influence probabilities are provided in advance as input. Other models proposed in the literature compute the probabilities through mining the users' past behaviors. Tang, et al. [113] studied topic-based social influence. In these social networks, discussion topics are distributed across users. The problem is to find topic-specific subnetworks, and topic-specific influence weights between members of the subnetworks. They propose a graphic probabilistic model called a Topical Factor Graph (TFG) to unify the information in one probabilistic model. Then they proposed the Topical Affinity Propagation (TAP) model which uses TFG to infer the influence graph. They also dealt with the efficiency problem by devising a distributed implementation of TAP.

Saito, et al. [127] studied the problem of building influence from users' past actions. They focused on the independent cascade model of influence. They formally defined the likelihood maximization problem and then applied an Expectation Maximization (EM) algorithm to solve it. Their formulation dose not however scale to larger data sets like social networks. This is due to the fact that in each iteration, the EM algorithm must update the influence probability.

Other Influence Models

There are many influence models that are based on greedy algorithms. Nemhauser, et al. [128] shows a greedy approximation algorithm to address the problem of finding a maximal set of individuals. Kempe, et al. [3] also proposed a greedy algorithm, but it suffered from an efficiency problem because the model needs to execute Monte-Carlo simulation several times until it provides accurate results which leads to very long computational times. There are studies concerned with improving the efficiency of greedy algorithms to maximize the influence, such as [129, 130].

Leskovec, et al. [129] studied the influence problem from a different perspective. The main question in their study is how to select nodes in a network to detect the spread of a virus as soon as possible? This is called outbreak detection. They developed an efficient algorithm based on 'lazy-forward' optimization. The algorithm was optimal and 700 times faster than a simple greedy algorithm. However, the approach still faces problems related to scalability. Chen, et al. [130] improved the efficiency of the greedy algorithm.

3.4 Review of Existing Approaches

In social networks, nodes that adopt an idea or a behavior are called active, while the other nodes which are not affected by this idea or behavior are called inactive. Based on this definition, the problem of influence maximization in social networks is based on finding the smallest number of nodes k that can influence the highest number of other nodes in the social network [31]. Although many models have been proposed to address the problem of measuring influence probabilities a node has over other nodes in the social network, they exhibit limitations in identifying the most influential users and the influence propagation process.

Static influence models [123] are very basic influence observation methods in social networks which are time-independent and based on fixed probabilities associated with the nodes in the network. These types of influence models do not diffuse over time since they assume that influence probabilities are fixed (i.e. static) and thus the influence propagation pattern does not change over time. In other words, these models do not consider the dynamic and the potential changes in the influence spectrum over contemporary social networks. One of the probabilistic techniques used in such models is based on Bernoulli Distribution, whereby a user tries to influence its inactive neighbors with the same probability. Static influence models are easy to apply to measure influence in a social network. But since social networks are dynamic in nature and therefore change over time, ignoring these changes makes them an appropriate practical choice to measure influence in today's social networks. To address this drawback, dynamic influence models were introduced. These influence models consider the dynamic valuation of influence probabilities associated with the network nodes over time. Examples of such models are snapshot and ordinal-time models. The snapshot model [124, 35, 122] takes various numbers of snapshots of the network at different successive timestamps to generate an evolving observation about the network. The Snapshot technique is widely used to model social networks because it can capture large-scale data and make it available for further analysis, particularly to measure influence progression over time.

On the other hand, this process of continuous observations needs to generate many snapshots, which can result in a lot of data. Ordinal-time models consider a time sequence in an evolving social network where each time an individual adopts a new behavior and a new network link is formed. This moment-by-moment measurement, though appealing, makes these ordinal-time models less practical to implement on large scale networks, and thus limits their ability to analyze the temporal evolution in adoption behaviors [2].

To address the issue of influence propagation, diffusion influence models such as linear threshold and cascades independent models were introduced. In linear threshold [31, 3, 125], every node contributes a certain weight to its adopting neighbors. If the sum of these weights is greater than a given threshold, the node becomes an adopter too. The weight depends on the edge strength between the node and its neighbors. Using the weight as a measurement between the node and its neighbors reveals the strength of the influence. In independent cascade models [3], each node has two states, to adopt or not to adopt the behavior. Influence propagation is measured using cascade processes where the adopters will have influence on their neighbors and the adopter neighbors will have influence on their own neighbors too, and so on. The influence spreads over the network as a result of these cascade sequences. Each adopting node has one chance to influence its neighbor to adopt the same behavior with a probability that depends on the edge strength between the node and its neighbor. These models are fast in spreading behaviors between nodes, especially when the initial set of the most influential nodes in the network is determined. Both linear threshold and independent cascade models do not reconsider the correlation between users' actions, and ignore alternative influence propagation methods to achieve adoption at a later activation instance. This raised the need for models that measure influence based on the dynamic nature of users' behaviors and contexts. The topical affinity propagation model [113] uses a topical factor graph (TFG) to build the influence probability model based on the users' topics. This model computes the influence probabilities through mining the users' behaviors and employs a distributed machine-learning algorithm to deal with the efficiency problem.

Sun and Tang [119] conducted a survey of social influence analysis models and algorithms. They discussed correlations between social similarity and influence and, in doing so, they focused on the computational aspects of social influence analysis by determining the selection of people who are similar to one another (for example two users who have the same opinion). They argued that people tend to influence other people who are already similar to them (for example changing the opinion of a user to agree with one of his neighbors). They also provided methods to measure the weight of influence.

However, novel applications of artificial intelligence techniques such as fuzzy logic could further contribute to maximizing influence in social networks. In chapter 4, we present a technique which will be further developed in a subsequent chapter.

3.5 Classification of Influence Propagation Algorithms

In this section, we compare the influence models discussed in the previous section as compiled in Table 3.1. Static influence models are based on capturing influence at the current moment. They are time independent models that do not diffuse over time. They assume that influence probabilities are fixed (static) and do not change over time. Different techniques are used in static influence models one of them is Bernoulli Distribution. Static influence models are easy to apply and test which makes them one of the easiest ways to measure influence in a network. But since social networks are dynamic, where new links are built/removed regularly, then the assumption is that static influence models will not be the best choice to measure influence in social networks. Dynamic influence models were introduced to address static influence probability deficiency. Based on these dynamic models, influence probability changes over time. Snapshot and ordinal-time models are discussed as types of dynamic influence models that are time-dependent. Snapshot models take different snapshots of the networks and generate an observation about the network. The snapshot technique is widely used to model social networks because it can capture large-scale data. To get a better observation of the network, we need to take many snapshots for large data sets, which is time consuming and needs a lot of space. Ordinal-time models provide detailed temporal dynamics of the network. They provide more accurate results since they measure influence moment-by-moment. There is no direct implementation of ordinal-tTime Models on large data sets (such as large social networks), which makes it difficult to draw conclusions about these models on social networks. Diffusion influence models such as linear threshold and cascades independent models were introduced to address the issue of influence propagation. In linear threshold, every node contributes a certain weight to its adopting neighbors. If the sum of these weights is greater than a given threshold, the node becomes an adopter too. The weight depends on the edge strength between the node and its neighbors. Using the weight as a measurement between the node and its neighbors will show the strength of the influence. Cascade independent models use cascade processes to measure influence propagation. Each node has two states, to adopt or not to adopt. The adopters will have influence on their neighbors and the adopter neighbors will have influence on their neighbors too, and so on, as the influence spreads over the network. Each adopting node has one chance to influence its neighbor to adopt the same behavior with a probability that depends on the edge strength between the nodes. These models have the advantage of fast spread of information through the nodes, specially when determining the initial optimal set of the most influential nodes in the network. Both linear threshold and cascade independent models have the same limitations since they both ignore the attributes associated with each user node and do not consider the correlation between user actions.

Other models based on user behavior were used to measure influence. The topical affinity propagation model uses TFG to build the influence probability model based on the user's topics. This model employs a distributed learning algorithm to deal with the efficiency problem, but it cannot capture the social influence between users while building a unified probabilistic model. We discussed also models that are based on greedy algorithms and we found that they outperform influence measures based on the structure of the social network, such as degree and distance centralities. However, on the other hand, their efficiency is low since they take a long time to execute tests repeatedly to provide accurate results.

3.6 Experiments and Results

In this section we will compare influence spread based on two famous influence propagation models the linear threshold (LT) model and the independent cascade (IC) model and we will propose our own model of influence named as the 'Similar Independent Cascade'. For estimating influence weights we proposed a method called the 'Jaccard Coefficient Based on Common Actions'. Then we compared the influence spread of our method against other methods. Linear threshold and the independent cascade algorithms, using Matlab were implemented. Then we conducted experiments using an Apple iMac with Mac OS X version 10.6.8, processor 2.66 GHz intel Core i5 and 4GB memory.

The experiment was applied to two real world social networks. The first social network is Flickr, which is a photo sharing social network. On Flickr, users can share and embed photographs on their own blogs. The dataset for Flickr consists of 2,570,535 nodes and 33,140,018 links between the nodes. We used the dataset provided by Meeyoung Cha, Alan Mislove and Krishna P. Gummadi [85]. 500 nodes were selected. These nodes are associated with actions to select a photograph. The second social network was Last.fm[2], which is a popular Internet radio to stream music. The dataset provided by Cantador, et al. [131] contains 1892 users who assigned tags to artists during different timestamps. A tag could be any word related

[2]http://www.lastfm.com

Models		Based On	Techniques	Algorithms	Strength	Limitation
Static Influence Models [123]		Capturing the most influential node in the current time	Bernoulli distribution	Maximum Likelihood Estimation (MLE)	Easy to apply and test	Does not consider the dynamics of social networks
Dynamic Influence Models	Snapshot [124, 35, 122]	Taking various number of snapshots for the network	Snapshots	Observations of the social network	- Can capture large-scale data and provide them for analysis - More applicable and easily handled	Need to take many snapshots for large size data
	Ordinal-Time	Detailed temporal dynamics of the network	Moment-by-moment measure-ment	Expectation Maximization (EM)	Provide more accurate influence results	No direct implementation of the ordinal-time definition on large-scale data
Diffusion Influence Models	Linear Thresh-old [31, 125, 3]	Choosing a threshold at random.	Threshold	-Approximation algorithms -Greedy algorithm	- Adopting new behavior based on fraction weight	- Does not consider the correlation between users' actions. - Ignores the attributes that are associated with each user node
	Independent Cascade Model [3]	Activating nodes based on discrete steps	Cascading processes	-Approximation algorithms -Greedy algorithm	- Fast spreading of behaviors	- Does not consider the correlation between users' actions. - Ignores the attributes that are associated with each user node
Models of influence based on users' behavior	Topical Affinity Propaga-tion (TAP) [113]	Computing the probabilities through mining the users' behavior	Graphical probabilistic model	-Affinity propagation algorithm. -Distributed learning algorithms	Devise a distributed learning algorithm under the Map-reduce programming model (deals with the efficiency problem)	TFG model can not capture the social influence between users while building the unified probabilistic model

Table 3.1: Comparison of Social Influence Models

to the artist like rock, POP, sad, touching, etc. The timestamp shows when the tag assignments were done. We selected 99 nodes. Each user is associated with his/her action of tagging an artist.

To compare the two influence propagation models we used four methods to assign edge probabilities in the social graph:

- Jaccard Coefficient Based on Common Actions: we proposed this method where we calculate the similarity between two nodes based on the common actions they have. The Jaccard Coefficient measures the commonly active properties of nodes u and v to the number of active properties in u or v. The formula used is $JC_{u,v} = \frac{A_{u,v}}{A_u + A_v - A_{u,v}}$, where A_u is the number of actions performed by node u, A_v is the number of actions performed by node v and $A_{u,v}$ is the number of common actions performed by nodes u and v. Algorithm 3.1 shows the steps of calculating the common actions between two nodes u and v.

- Weighted Cascade (WC): which is a special case of the independent cascade model where each edge from node u to v is assigned a probability $\frac{1}{d_v}$ of activating v [3].

- Trivalency (TV): where edge probabilities are selected uniformly at random from the set {0.1, 0.01, 0.001}

- Uniform (UN): where all edges have the same probability (e.g. $p = 0.01$)

Algorithm 3.1 Jaccard Coefficient Based on Common Actions Algorithm

```
1. Find all $actions_u$;
2. Find all $actions_v$;
3. For $\forall a \in actions_u$ do
    (a) if $a$ is in $actions_v$ AND $time_a < time_v$do
        i. $common_{u,v} = common_{u,v} + 1$;
4. $JC_{u,v} = \frac{common_{u,v}}{actions_u + actions_v - common_{u,v}}$
5. return $JC_{u,v}$
```

Using the Flickr social network we noticed that the independent cascade model outperforms the linear threshold model in all probability assignment methods when the seed set size becomes larger. Across a range of influence valuation methods, initial results show that our approach applied to the independent cascade model outperforms an existing landmark influence propagation model called the linear threshold model. The common actions (Figure3.12(a)) probability assignment methods is steadier and both propagation models provide similar curves, although it does not activate as many nodes as other methods; shown in Figure3.12(b)(c)(d). Results could be different for different sittings applied to run the algorithms such as the threshold value or the number of nodes in the seed set.

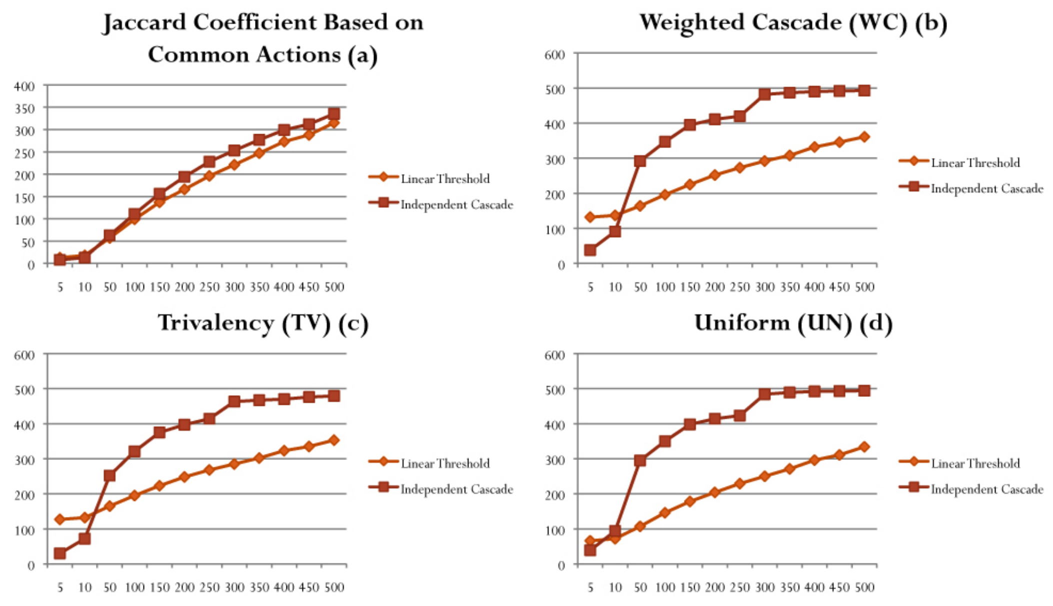


Figure 3.12: Comparing Influence Spread For Flickr Social Network

Using the Last.fm social network, we applied the same probability assignment methods used above on the dataset. In this experiment we noticed that in the trivalency and uniform methods the independent cascade model outperforms the linear threshold model by activating more nodes during the propagating process Figure 3.13.

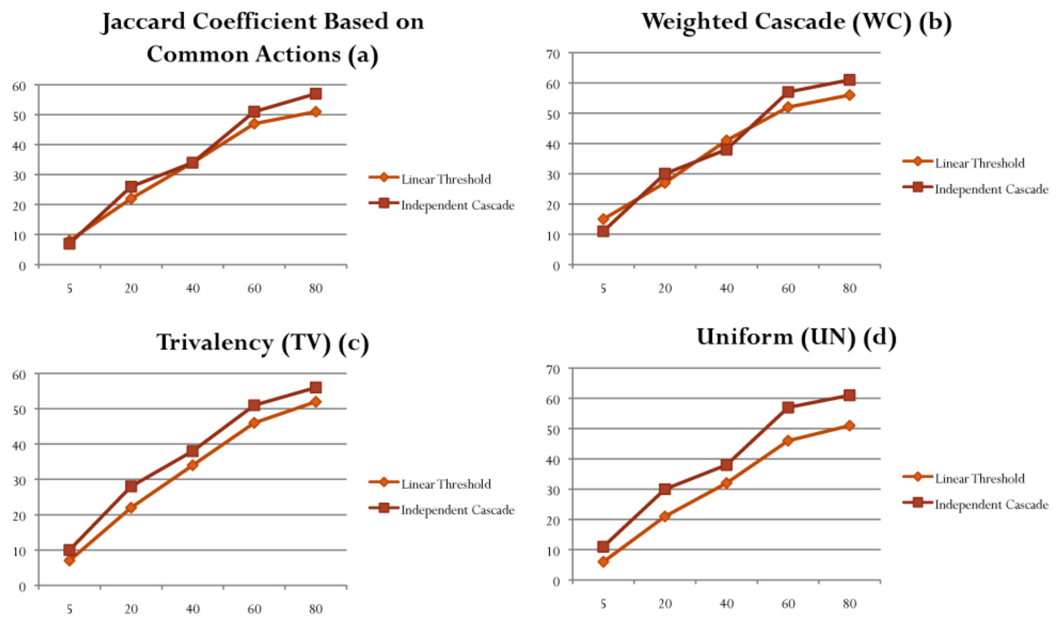


Figure 3.13: Comparing Influence Spread For Last.fm

3.7 Summary

In this chapter, we discussed the metrics used to measure influence probability. We introduced some new metrics to analyze the centrality in social networks. We also surveyed state of the art research which addressed the objective of influence maximization in social networks. We highlighted the strengths and limitations of these approaches through a comparative study and focused on two landmark propagation algorithms to assess their diffusion performance in real-world networks. We made comparisons using different methods to assign edge probabilities in the social graph, including the proposed method of estimating influence using the Jaccard Coefficient based on Common Actions. We found that independent cascade outperforms linear threshold in every experiment and thus we used it as a candidate for performance evaluation against our proposed algorithm (see the next chapter).

Chapter 4

Community Aware Influence Maximization

4.1 Introduction

Communications and recommendations started a long time before the internet. Users communicate with each other in order to seek opinions from their friends about specific products. Then friends will give their opinion directly about the product. The user will make up his mind to purchase or not based on the suggested opinion of his friend. In later years the Internet has been used to provide users with an opportunity to surf the web for products. Users can use their computers to search the web for a product, but the internet is a huge repository that has too products of different styles and shapes, which makes it difficult to chose. Currently new websites like Twitter and Facebook have become the most visited websites on the Internet. These social websites provide users with a richer communication experience. For example, if a user wants to watch a movie with his friend, and that friend likes a specific type of movie, then one of the first things a user will do is to go to his Twitter account and post a tweet to seek the suggestions of his/her friends. He/She might have different connections with his/her friends. Some with strong ties and some with weak ties. Strong ties means high trust and the willingness to accept suggestions, while the user might be hesitant to accept the suggestions from other friends who have weak tie connections. Those friends will recommend movies based on their personal opinion and respond by tweeting their recommendations. Some of these friends might also get the recommendation about the movie from their friends or friends-of-friends. So social ties play an important role nowadays. From the previous example we can see that there are some considerations that encourage a user to accept (adopt) a recommendation or recommend a products to his friends such as 1) number of friends who adopted that behavior. The more your friends talk about a product, the more curious you become about it. The more curious you want to try that product. Also 2) how strong is your relationship with the friend who recommended the product. The stronger the relationship, the more trust you will have, the more influenced you become by that friend's opinion. 3) The type of your friends and their references. If you go out with a friend who likes action movies you will probably try to find a movie that matches his/her preferences and you will probably be interested in such a movie genre because your friend likes it.

Nowadays users are overwhelmed by many products, and due to the fast-pace of life they have limited time to invest in the process of searching and finding other options. So the solution is to combine Internet resources with social communities. These connections between users will make it easier to send and accept recommendations. This is because people tend to accept ideas and product recommendations from their friends. Social connections are also important for business. As companies want to advertise their products with minimal effort and a minimal budget. They want to increase sales and create new revenue sources and also they want to solve the 'cold start' problem, when a company has new products and the system cannot derive any inferences for the users, because the system has not gathered enough information yet. So using 'viral marketing' with social connections, such as selecting a user who has high connections and high influence in his community, can spread knowledge about products through the social network. The aim of finding such individuals is referred to as influence maximization [31]. This problem is becoming one of the most demanding issues in the current age of social networks.

Many companies are currently using social networks as a base for their marketing and promotional campaigns, as social networks provide an easy cost effective way to spread knowledge about a product.

In this chapter, we address the influence maximization problem where we strive to find a set of k nodes that can spread influence to the largest set of other nodes in the network. For that purpose we adopt a two-phase approach. First we identify synthetic communities within the network which results in subsets of similar nodes. Then, we discover ‘key nodes’ within each virtual community to represent the seed of influence propagation. Hence, our community-based influence propagation algorithm starts by detecting communities in the social network as a pre-processing step to group similar nodes together. This process will emphasize the influence due to the similarity between the nodes and the similar behaviors they will perform. After grouping similar nodes together, we identify the set of ‘key nodes’ that contains the most influential nodes within each virtual community. The rationale of this approach is that similar nodes tend to behave similarly, and hence influence is embraced faster as similar nodes tend to interact and adopt each other’s behavior. We use centrality measure to uncover influential nodes. However, this attribute is based solely on network structure and ignores individual nodes’ influence weight. Therefore, we combined centrality degree with an estimated weight measure which we derive based on common actions between two nodes. We propose to harvest historical action logs between nodes to elicit an estimated influence weight based on their common actions. This value dynamically changes based on the behavior of nodes in the social network. To evaluate the degree of influence, and decide which nodes are most influential, we use a fuzzy logic inspired approach to select the influential nodes based on both their central location and influence weight. Finally, our method selects the final set of influential nodes that we call a ‘seed set’, where members are predicted to have the highest propagation rate across the network. Such research works to discover how the most influential users can help in performing different activities around OSNs such as targeted advertisements, and item recommendations to OSNs users. Research carried out in this area is sparse, and spans multiple disciplines.

The rest of this chapter is organized as follows: Section 4.2 provides background and related work relevant to our proposal on influence maximization based on common actions and fuzzy logic. Section 4.3 describes the community-based influence propagation algorithm. Section 4.4 reveals our experimental findings, which resulted from applying the algorithm to real-world online social networks. Finally, Section 4.5 concludes the chapter with a summary and some possible future directions for this social influence model.

4.2 Fuzzy Logic Inferences in Social Networks

Fuzzy logic was introduced by Lotfi Zadeh in 1965 [132]. Fuzzy logic is a type of probabilistic logic that allows medium values between 0 and 1 to be considered for reasoning about qualitative situations with gradual levels of affirmation (i.e. yes/no) or truthfulness (i.e. true/false)[133]. The reasoning process is approximate rather than fixed or exact. This kind of reasoning is more like human thinking [134] and is more logical in judging natural situations. We are using a fuzzy logic inspired method in our approach to determine the nodes of a social network with high probability to join the membership of 'key nodes' used as a seed for influence propagation.

Fuzzy logic has the unique feature of being a simple and flexible human language rule-based approach [135]. A fuzzy logic based system converts these rules into their mathematical equivalents, which provide more realistic behavior in real world situations [136]. Unlike standard logic methods, the truth of a specific situation can be a range of values. These fuzzy systems need a membership function that clarifies how to calculate the correct value between 0 and 1 to match a given affirmation [137]. This can be represented by a value between 0 and 1. The human language rules and the ability to use membership functions make such a system easy to update and to maintain.

We will give an example inspired from [138] and based on our method of finding influential nodes. For example, let's assume that we want to find the most influential nodes in a social network. Given a social network of five nodes, $nodes = \{1,2,3,4,5\}$ we want to find the nodes that will be the most influential, given the constraints that the node should be in a central location and it have a high influence weight on other nodes in the network. We need to represent the first constraint using a fuzzy set $Centrality(C) = \{\{1,0.4\},\{2,0.6\},\{3,0.8\},\{4,0.4\},\{5,0.5\}\}$ this shows that node 3 has the highest membership grade, which means that node 3 has the highest centrality in this network. While node 1 has the lowest membership function of 0.4. Then we represent the second constraint in a fuzzy set $InfluenceWeight(IW) = \{\{1,0.1\},\{2,0.9\},\{3,0.7\},\{4,1\},\{5,0.2\}\}$. In this fuzzy set the membership grades indicate the average influence weight of a node in the whole social network. The highest means the most influential in the network. From that set we notice node 4 is the most influential while node 1 is the least influential in the whole network. After representing all constraints as fuzzy sets, we need to make a decision which of these nodes is the most influential. For that we apply the standard function fuzzy intersection that helps in making the fuzzy decision. It can be thought of as combining the constraints together to come up with the best overall decision[138]. The fuzzy intersection between two fuzzy sets is computed by taking for each element (nodes in our example) the minimum of its membership in both sets [137]. Thus the minimum of the two fuzzy sets $Centrality(C)$ and $InfluenceWeight(IW)$ is given by $C \cap IW = \{\{1,0.1\},\{2,0.6\},\{3,0.7\},\{4,0.4\},\{5,0.2\}\}$. Figure 4.1 shows a graphic representation of the fuzzy decision. From the figure we find the most influential nodes by looking at the node with the maximum membership grade. In our example and as shown in Figure 4.1, we noticed that node 3 seems the most influential node in the network based on its location and influence weight.

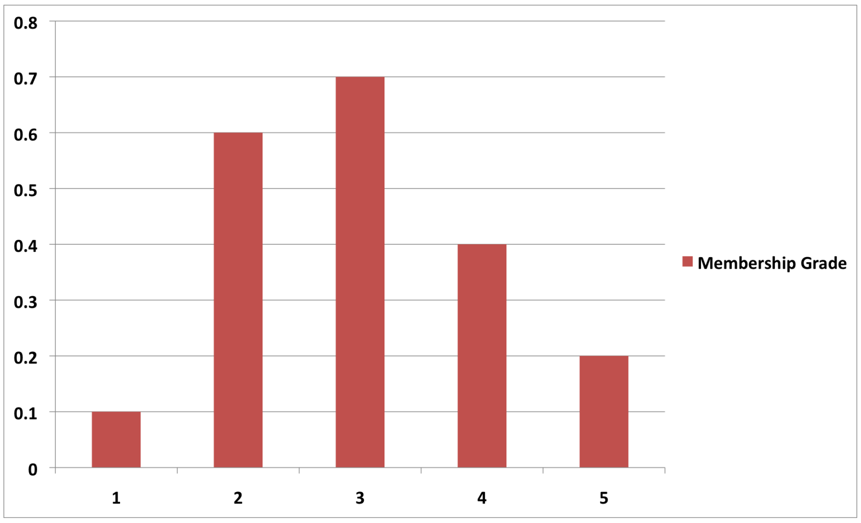


Figure 4.1: Fuzzy Decision Plot

4.3 Community Aware Influence Propagation Using Fuzzy Logic Inferences

In this section, we suggest our community detection and influence maximization algorithm. The objective of which is to maximize influence and increase the activation of nodes in a social network. In this approach, we combine the previous techniques which were introduced, namely community detection [139] and influence weight calculation [140]. We anticipate from this approach to identify the seed set of users who can influence their neighbors to behave similarly. Our experimental evaluation on real-world networks confirms this statement.

Our proposed algorithm, shown in Algorithm 4.1, consists of three main steps:

1) Detecting communities (Step 1 in Algorithm 4.1)

2) Identifying key users in each community (Step 4 in Algorithm 4.1), and

3) Finding the seed set to propagate influence across the entire social network (Steps 5 and 6 in Algorithm 4.1).

Figure 1.5 illustrates our proposed framework.

Algorithm 4.1 Community Aware Influence Propagation Algorithm

```
1. Detect Communities $C$ of ($G$) (Algorithm: 2.8).
2. $Threshold\_IU \leftarrow$ Number of Important Users.
3. $Threshold\_S \leftarrow$ Number of Users in the Seed Set ($S$).
4. For each community $C$ do :
   (a) Find Central Users Fuzzy Set
       i. $CentralUsers \leftarrow$ Find Central Users (Algorithm: Degree Centrality)
       ii. For each $CentralUsers$ do
           A. $CentralityWeight = nodeDegree/totalEdges$ //Membership Function
   (b) Find Influence Weight Fuzzy Set
       i. For each $CentralUsers$ do
           A. InfluenceWeight $\leftarrow$ Calculate Influence Weight (Algorithm: 3.1)
       ii. For each $CentralUsers$ do
           A. $InfluenceWeightsAvg = sumInfluenceWeights(node)/totalNodes$
              //Membership Function To Calculate Average Influence Weight
   (c) Find The Important Users (Fuzzy Decision)
       i. $Intersection = min(CentralityWeight, InfluenceWeightsAvg)$
       ii. $ImportantUsers \leftarrow$ Select The Maximum Intersection Grade To Decide The Important Users
5. For each $ImportantUsers$ do
   (a) Reachability $\leftarrow$ Apply Influence Propagation Method (Algorithm: 4.2)
6. $S \leftarrow$ Select The Top Influential Users Based on $Threshold\_S$
7. Return $S$
```

4.3.1 Detecting Communities

Starting from a social network, the first step of our framework consists in detecting communities. Finding communities is one of the most important steps that we must perform. This step determines the success or failure of the adoption of behaviors in the social network, because of the similarity factor within communities that is used for influence propagation. Well structured communities will facilitate a better dissemination of influence. Therefore, the role of this initial community detection step in the influence propagation algorithm is important. Similar users tend to adopt similar behaviors, and detecting these communities in the network will make it easier to find the key nodes in each community to form the seed set for the whole network. Algorithm 4.1 reveals an enhanced version [139] of the CNM algorithm proposed by Clauset, et al. [48]. Our experiments on Similarity-CNM showed that pre-processing social network data and considering other factors related to the network structure can optimize the community structure. Based on this observation, starting the influence propagation algorithm by dividing the social network into virtual communities of similar users is poised to maximize the process of influence spread within the social network, as members of the same community tend to behave similarly. Next, we discuss Algorithm 4.1 where the quest for key users starts by finding central users and then determining those with higher influence weights.

4.3.2 Finding Key Users

After dividing the network virtually into groups of similar communities, we then identify the initial set of key users in each community. These users are potential candidates of the final seed set in the network. There are two main characteristics of a key user: 1) location in the network (centrality), and 2) historical influence activity. This is a multi-criteria decision making problem for which fuzzy logic can provide a solution [141]. Indeed, multi-criteria decision making problems consist of (1) a finite set of criteria (or properties) that evaluate the quality of a key user to join the seed set of users used in our influence propagation model, and (2) weights (or importances) of the criteria [142]. These decision-making components map to our problem to identify key users where the criteria are represented by centrality and influence power, and weights are represented by the level associated with these criteria. Based on these criteria and weights, a fuzzy logic process is employed to find key users, and hence the seed set of users who will drive influence propagation, as shown in Algorithm 4.1.

Finding Central Users Fuzzy Set

The first main characteristic of a key user is to have a favorable location in the network. For example, central nodes have high connections with other users. Also nodes that reside between two groups play the role of a bridge to convey behavior from one group to another, and so they would be good key user candidates. We used a degree centrality measure to determine users with favorable locations. These central users are elected as members of a fuzzy set which accumulates users with favorable locations based on a membership function which is discussed below.

The process starts by calculating the degree centrality of each community user. This is done by calculating the in-degree and out-degree of each node then summing up these values. A centrality threshold is then used to discriminate the central nodes for our algorithm's further consideration. Nodes that have higher in and out degree than the threshold are considered central in the social network. The membership function calculates the degree weight of all central users based on the following formula:

$$CentralityWeight = nodeDegree/totalEdges \tag{4.1}$$

Where *nodeDegree* is a variable that stores each node's in and out degree, *totalEdges* is the total number of edges in the network. This formula calculates the centrality weight of each node's in and out degree based on the total number of edges in the network. The resulting values vary between 0 to 1, which reflect the centrality level of each node in the network. Whenever the result is approaching 1, this means the node is probably in a very central location. We will use this value later to make a decision about whether the node could be a key user.

Finding Influence Weight Fuzzy Set

The second main characteristic of a key user is its ability to influence many individuals. To calculate influence weights, we propose the integration of Jaccard Coefficient Based on Common Actions technique [140]. To illustrate this technique, consider a user A who started a behavior at time $T1$, and after a while another user B adopts the same behavior at time $T2$. This means B is influenced by A after a certain time. For the Jaccard Coefficient Based on Common Actions technique, we find out the number of similar actions a user has embraced after his friend had adopted the same behavior. We assume there is only one source for each action. This assumption is actually validated by the sample data set used in our experimental evaluation.

After calculating the influence weights, we propose to calculate the average influence weight to discriminate nodes with the highest historical influence activity in the network. Finding the average can be calculated through the following membership function:

$$InfluenceWeightsAvg = sumInfluenceWeights(node)/totalNodes \tag{4.2}$$

Where $sumInfluenceWeights(node)$ is the sum of all influences weights that a specific node has on every node in the network; $totalNodes$ is the total number of nodes in the social network.

Fuzzy Decision Making

Key users are determined based on a Fuzzy Set Intersection value for each node n using both $CentralityWeight_n$ and $InfluenceWeightsAvg_n$ criteria weights, as shown in the following formula:

$$Intersection_n = min(CentralityWeight_n, InfluenceWeightsAvg_n) \tag{4.3}$$

Intersection determines the lowest of the two values between the centrality weight of a node (*CentralityWeight*) and the average weight of influence for that node (*InfluenceWeightAvg*). Then, we select the maximum membership grade generated from the above intersection process to decide on key users. To illustrate this step, we anticipate each node to have a defect in either centrality or its influence value, so we select the lowest value of these two values to minimize the effect of either deficiency, and then compensate this deficiency by detecting the nodes with the maximum of these lowest values. These nodes will be the least defective of all other nodes and they will definitely be the best key user candidates. Equation 4.4 shows a mathematical representation of this process.

$$MembershipGrade_{max} = max(Intersection_n)\ \forall n \in CN \tag{4.4}$$

Where *CN* is the set of central nodes in the social network.

4.3.3 Finding Seed Set

After finding the key users in each community, we select the top ones which have highest influence propagation in the social network to form the seed set of nodes that are used to diffuse influence across the network. Algorithm 4.2 shows our influence propagation method which is based on an independent cascade propagation model. Our algorithm takes into consideration the influence power that a node might have on other nodes indirectly. This means a node might activate nodes other than its direct neighbors. This is inferred through historical common actions as a method to estimate weights probabilities. The complexity of this approach is similar to the IC model since the propagation process uses the same steps but the nodes selected for propagating the influence are different in our approach from the IC one (see

Algorithm 4.2).

Algorithm 4.2 Influence Propagation Algorithm

```
1. For $\forall u \in KeyUsers$ do
   (a) At step $t = 0$, activate $u \in KeyUsers$ and added it to $Coverage_0$
   (b) At each step $t > 0$, For $\forall u \in Coverage_{t-1}$ do
       i. For $\forall v_{inactive}$ if $InfluenceWeight_{u,v} \geq InfluenceThreshold$
          A. Activate $v$
          B. $ActiveList = ActiveList \cup \{v\}$
          C. $TotalCoverage = TotalCoverage + 1$
       ii. All the nodes activated at this step are added to $Coverage_t$
       iii. This process ends at a step $t$ if $Coverage_t = 0$ /*no more nodes to
            activate*/
2. Add nodes $u$ with highest $TotalCoverage$ to $S$
3. Return $S$
```

4.4 Experiments and Performance Analysis

In this section we discuss our experimental environment and reveal the results obtained from our proposed approach.

4.4.1 Experiment Environment

As an experimental platform, we used Flickr real-world social network dataset. Flickr is a photo sharing social network. On Flickr users can share and embed photographs in their own blogs. We used the dataset provided in [85], which consists of 2,570,535 nodes and 33,140,018 links between the nodes. Due to computational constraints and as part of our preliminary experiments, we randomly selected 500 nodes, to run our experiments. We are planning to increase the size of the sample data set in the future to run further experiments.

Algorithm 4.1 was implemented using Matlab and C++. The experiments were performed on an Apple iMac with Mac OS X version 10.6.8, processor 2.66

GHz intel Core i5 and 4GB memory.

4.4.2 Candidate Algorithms and Performance Metrics

We compared our proposed algorithm to the benchmark independent cascade model. The algorithm of the independent cascade model starts with an initial set of active nodes A_0. This set of individuals should be chosen to generate maximum influence during the cascade diffusion process. On the other hand this model does not consider the correlation between users' actions, which we we considered in our modified approach of independent cascade model depicted earlier in Algorithm 4.2, where we exploit those social connections and similar actions from different users.

As an evaluation metric, we used the number of activated nodes to show the performance of our algorithm. We focused on finding the number of activated nodes because it is the main factor to compare the propagation of influence for both the independent cascade (IC) model and our proposed model. Node activation can be explained as embracing a certain behavior by a node that is initiated from another node.

4.4.3 Results and Discussion

Through our experiments, we noticed interesting results generated by our proposed method. Compared to IC model, our community-aware social influence model (discussed in Algorithm 4.1) activates higher number of nodes starting from a smaller seed set (compared to the IC model).

Figure 4.2 shows the results generated by both candidate algorithms when the seed set maximally contains 5 nodes. Based on the original propagation of the IC model, the 5 seed set nodes will activate 33 nodes in the social network. While in our approach, the same number of seed set nodes activates 134 nodes. This shows that by using the social relations that are available in the network, more additional nodes are reached by the influence propagation process. In doing so, the initial nodes are more successful in persuading neighbors or neighbors-of-neighbors to adopt the propagated behavior.

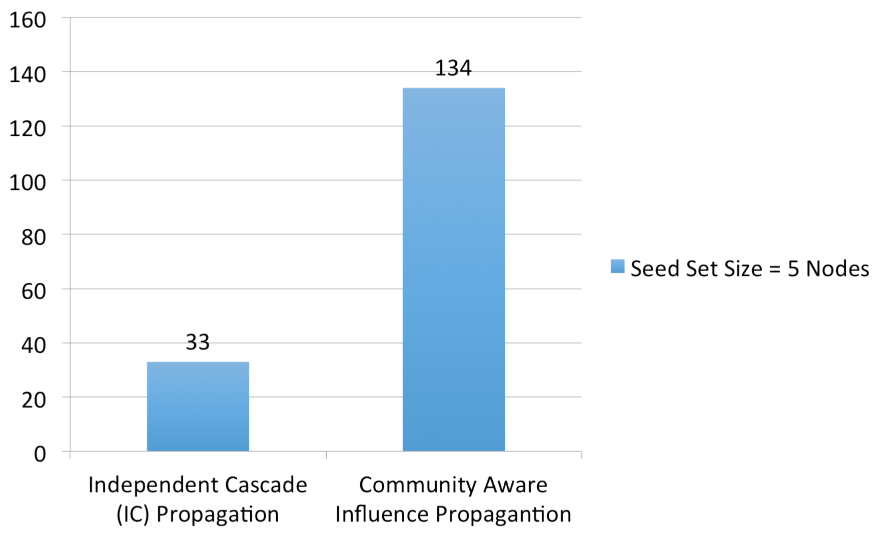


Figure 4.2: Activated Nodes When Seed Set = 5

In another experiment, we increased the size of the seed set to 30 nodes. Using the IC model, 30 nodes in the seed set activate 98 nodes in the social network as shown in Figure 4.3. On the other hand, in this experiment our approach first discovers the top 6 nodes out of the 30 nodes and then selects them as potential candidates for the seed set. The 6 nodes activate 135 nodes in the social network. This shows that our approach can select the minimal number of influential nodes as seed set members who can activate a larger number of nodes in the social network. The seed set in our approach contains fewer nodes compared to the IC model startup nodes, and at the same time our approach activates more nodes in the social network. For example, if there is a company that wants to promote a product using our approach, they can convince fewer numbers of initial people to promote their product. This way they will save on the expenses of having an advertisement campaign or save on the expenses of providing these products for free to those initial people (seed set members). At the same time, the company will generate more revenue, as those initial people are capable of persuading many other people in the social network who are expected to embrace the product later on. We noticed more interesting results by changing the threshold $InfluenceThreshold$ of similarity probability used as weight of influence propagation in Algorithm 4.2. When the threshold decreases, our approach reveals that 1 node can activate 144 other nodes in the network. By decreasing this threshold, we increase the number of nodes that are similar to the initial nodes. This means instead of finding 30 initial nodes we can find only 1 node which has connections that can activate about 29% of the total nodes in our sample social network as seen in Figure 4.4.

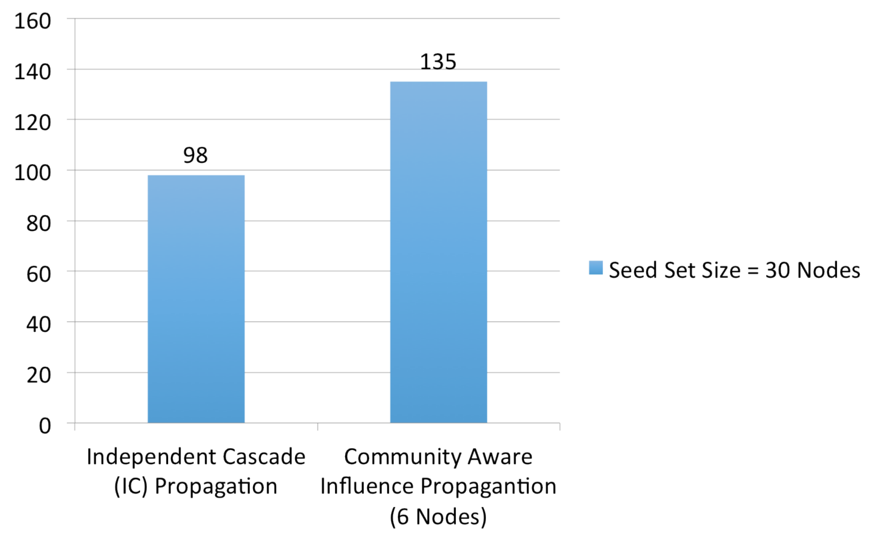


Figure 4.3: Activated Nodes When Seed Set = 30

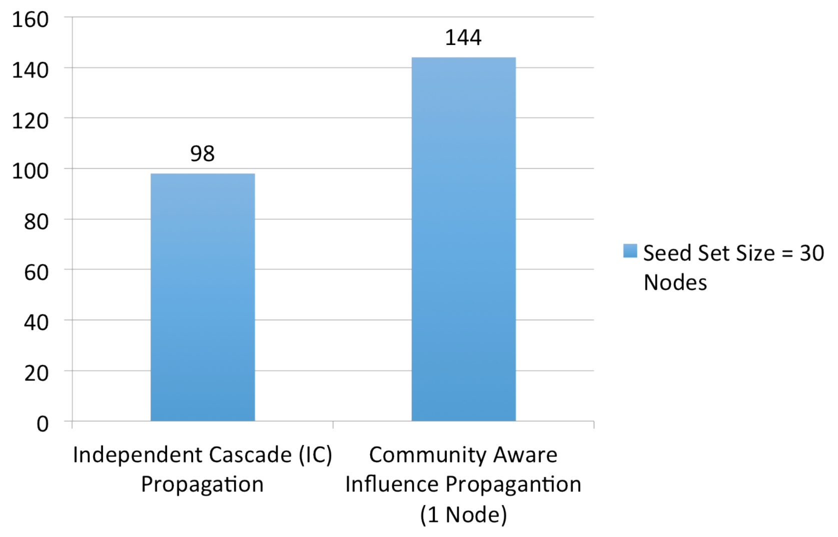


Figure 4.4: Activated Nodes When Seed Set = 30 and the Influence Threshold is Decreased

4.5 Summary

In this chapter, we discussed the components of our combined influence maximization framework which involved a community detection pre-processing step followed by eliciting key users in each community. We introduced a novel method of finding key users in each community based on a fuzzy logic method. We worked on each component of our framework separately then combined these building blocks to craft our overall model for influence maximization in online social networks. Our experiments on the real-world network, Flickr, showed more nodes activated than by using the benchmark independent cascade model.

Chapter 5

Conclusion and Future Work

In this chapter, we summarize the contributions of this dissertation and discuss some future research directions towards maximizing influence in online social networks.

5.1 Clustering For Intelligent Web

Clustering algorithms represent an important approach to divide and analyze data. There are many different types of clustering each with its own technique. We have discussed six clustering algorithms in this thesis: single link, average link, MST single link, K-means, ROCK, and DBSCAN. We discussed the accuracy issues involving these techniques when applied to the social web.

The single-link, average-link, and MST single-link algorithms are agglomerative hierarchical algorithms. They do not perform efficiently (both with respect to time and space) on large data sets even though they are easy to implement. K-means algorithm is a partitional algorithm, which is more efficient than link-based algorithms. However, it does not work with categorical data since it relies on the idea of centroids. Moreover it cannot handle outliers (the points that are far away from the main clusters) [58]. ROCK algorithm is a hierarchical agglomerative algorithm that can handle categorical data since it relies on the links more than the distance between nodes to cluster categorical data. However it has high time and space complexities. The DBSCAN algorithm is a density-based algorithm that uses point density to identify clusters in a space. It can handle outliers even though its time and space complexity are high.

The algorithms discussed in this thesis are used for the identification of groups of users and data. We propose to combine different algorithms in order to overcome their individual deficiencies. For example, K-means proved to be simple and quick since it can run on parallel computational platforms, and could be combined with other algorithms to overcome its weaknesses in addressing categorical data. This could maximize the benefits resulting from combining these algorithms in terms of better quality clusters [73]. One example would be combining the efficient K-means algorithm with the powerful ROCK algorithm (if the data is Boolean or categorical) or DBSCAN algorithm (if the data is spatial). One scenario would be using K-means on the high-level clusters then process them with the ROCK or the DBSCAN algorithm.

5.2 Enhancing Community Detection

Community detection algorithms are important to cluster data into communities and analyze their characteristics. However, different clustering techniques lead to different performance outcomes, and can become applicable to distinct real-world phenomena. CNM is a prominent community detection approach, which we proposed to improve through a pre-processing step, based on node similarity. We suggested two complementary approaches for that purpose: Similarity-CNM and ECD-Jaccard, to provide a better community structure. Similarity-CNM algorithm discovers similarity between nodes and builds a corresponding virtual social network. Similarly, ECD-Jaccard algorithm also computes nodes similarity as a pre-processing step, but then these values are assigned as weights to the network edges, resulting in a weighted virtual social network, unlike the Similarity-CNM approach (which is unweighted). The CNM algorithm is then employed to detect communities in both approaches. The experimental analysis reveal that these pre-processing techniques have an advantage over the original CNM algorithm in terms of community modularity. We have evaluated our methods on artificial networks and applied them to real-world networks. Simulation results show that our Similarity-CNM approach outperforms the original CNM algorithm by more than 50% in certain configurations of the network. The experimental results on the real-world network Flickr raises this performance scale to 67%. In ECD-Jaccard, the values of maximum modularity in weighted artificial networks outperforms the maximum modularity values in unweighted artificial networks by almost 11%.

Our work showed that pre-processing social network data and considering other factors related to the network structure can optimize the results generated by community detection algorithms. We plan to pursue our investigation to augment social networks with additional semantic information derived from their social aspects to further optimize community detection and related applications.

5.3 Social Influence

In this thesis, we defined social influence and stated its importance in evolving social networks. We introduced some analytics used to measure centrality in social networks such as degree centrality. We also surveyed influence maximization models in social networks. We stated the strength and limitation of each model through a comparative study. We compared two prominent propagation models; independent cascade and linear threshold models to evaluate their information diffusion performance. This original comparison used common actions influence weight estimation which has later been integrated in our proposed influence maximization algorithm. Based on performance evaluation grounds, we developed this comparative study to find the best candidate algorithm against which we evaluate our proposed algorithm. As independent cascade showed a higher performance, it was selected for further comparative analysis with our approach. In doing so, we also revealed some research directions in social network mining aimed at further analyzing social influence in order to generate more accurate results and help in addressing the limitations of existing approaches, such as scalability and efficiency.

5.4 Community Aware Influence Maximization Using Fuzzy Logic

We addressed the influence maximization problem and proposed a novel approach to increase propagation of influence in online social networks. Our results show the effectiveness of adopting a community-detection process prior to applying influence propagation techniques. We also proposed a novel method of discovering 'key nodes' in the social network based on a fuzzy logic inspired method. Using the proposed fuzzy-based technique to identify the most prominent nodes as an initial set for influence propagation provides a more dissuasive determination of the influential nodes, empowered by a combination of criteria such as the node's location and influential weight in the social network. There are many possible future directions to extend this influence maximization algorithm to address limitations such as scalability and efficiency. We are currently working on applying the algorithm on larger datasets to come up with a better view about the robustness of this algorithm in finding the most influential seed set. We are also studying the effectiveness of applying different centrality measures for better accuracy of result, such as with betweenness and closeness, etc.

Bibliography

[1] M. Ester, H. peter Kriegel, J. S, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise." AAAI Press, 1996, pp. 226–231.

[2] D. Cosley, D. P. Huttenlocher, J. M. Kleinberg, X. Lan, and S. Suri, "Sequential influence models in social networks," *In Proc. 4th International Conference on Weblogs and Social Media*, 2010.

[3] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '03. New York, NY, USA: ACM, 2003, pp. 137–146.

[4] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," *In IMC '07 Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pp. 29–42, 2007.

[5] G. L. Alexanderson, "Euler and königsberg's bridges: A historical view," *Bulletin (New Series) of the American Mathematical Soceity*, vol. 43, no. 4, pp. 567–573, 2006.

[6] D. B. West, *Introduction to Graph Theory*, 2nd ed. Prentice Hall, September 2000.

[7] S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1994.

[8] R. Pastor-Satorras and A. Vespignan, “Epidemic spreading in scale-free networks,” *Physical Review Letters*, vol. 86, no. 14, pp. 3200–3203, 2001.

[9] D. Boyd and N. Ellison, “Social network sites: Definition, history, and scholarship,” *Journal of Computer-Mediated Communication*, vol. 13, no. 1, pp. 210–230, 2007.

[10] Half of the world’s online population uses facebook. [Online]. Available: http://www.statista.com/topics/1164/social-networks/chart/1103/top-10-social-networks-in-q1-2013/

[11] D. M. Ehrlich, “Social network survey paper,” *International Journal of Learning and Intellectual Capital*, vol. 3, no. 2, pp. 167–177, 2006.

[12] A. laszlo Barabasi and J. Frangos, *Linked: The New Science of Networks*. Perseus Books Group, 2002.

[13] L. Zhen, H.-T. Song, and J.-T. He, “Recommender systems for personal knowledge management in collaborative environments,” *Expert Systems with Applications: An International Journal*, vol. 39, no. 16, Nov. 2012.

[14] R. I. Mooney and L. Roy, “Content-based book recommending using learning for text categorization,” in *Proceedings of the Fifth ACM Conference on Digital Libraries*, ser. DL ’00. New York, NY, USA: ACM, 2000, pp. 195–204.

[15] D. Billsus and M. J. Pazzani, “Learning collaborative information filters,” in *Proceedings of the Fifteenth International Conference on Machine Learning*, ser. ICML ’98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 46–54.

[16] J. S. Breese, D. Heckerman, and C. Kadie, “Empirical analysis of predictive algorithms for collaborative filtering,” in *Proceedings of Fourteenth Conference on Uncertainty in Artificial Intelligence*, ser. UAI’98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 43–52.

[17] J. Delgado and N. Ishii, “Memory-based weighted-majority prediction for recommender systems,” in *Proceedings of ACM SIGIR ’99 Workshop Recommender Systems: Algorithms and Evaluation*, 1999.

[18] A. Nakamura and N. Abe, “Collaborative filtering using weighted majority prediction algorithms,” in *Proceedings of the Fifteenth International Conference on Machine Learning*, ser. ICML ’98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 395–403.

[19] D. Y. Pavlov and D. M. Pennock, “A maximum entropy approach to collaborative filtering in dynamic, sparse, high-dimensional domains,” in *Proceedings of Neural Information Processing Systems*. MIT Press, 2002, pp. 1441–1448.

[20] M. Pazzani and D. Billsus, “Learning and revising user profiles: The identification of interesting web sites,” *Machine Learning*, vol. 27, no. 3, pp. 313–331, 1997.

[21] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, “Grouplens: An open architecture for collaborative filtering of netnew,” in *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*. Chapel Hill, North Carolina: ACM Press, 1994, pp. 175–186.

[22] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-based collaborative filtering recommendation algorithms,” in *Proceedings of the Tenth International Conference on World Wide Web*, ser. WWW ’01. New York, NY, USA: ACM, 2001, pp. 285–295.

[23] M. R. Subramani and B. Rajagopalan, “Knowledge-sharing and influence in online social networks via viral marketing,” *Communications of the ACM*, vol. 46, no. 12, pp. 300–307, December 2003.

[24] S. Yang and G. M. Allenby, “Modeling interdependent consumer preferences,” *Journal of Marketing Research*, vol. 40, no. 3, pp. 282–294, 2003.

[25] K. A. Falahi, N. Mavridis, and Y. Atif, *Computational Social Networks: Tools, Perspectives and Applications*. Springer, 2012, ch. Social Networks and Recommender Systems: A World of Current and Future Synergies, pp. 445–465.

[26] N. A. Christakis and J. H. Fowler, "The spread of obesity in a large social network over 32 years," *New England Journal of Medicine*, vol. 357, no. 4, pp. 370–379, 2007.

[27] E. M. Rogers, *Diffusion of Innovations*, 4th ed. Free Press, 1995.

[28] D. Strang and S. A. Soule, "Diffusion in organizations and social movements: From hybrid corn to poison pills," *Annual Review of Sociology*, vol. 24, no. 1, pp. 265–290, 1998.

[29] L. E. Blume, "The statistical mechanics of strategic interaction," *Games and Economic Behavior*, vol. 5, no. 3, pp. 387–424, 1993.

[30] H. P. Young, *Individual Strategy and Social Structure: An Evolutionary Theory of Institutions*. Princeton University Press, 1998.

[31] P. Domingos and M. Richardson, "Mining the network value of customers," in *Proceedings of 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '01. New York, NY, USA: ACM, 2001, pp. 57–66.

[32] J. Leskovec, L. A. Adamic, and B. A. Huberman, "The dynamics of viral marketing," in *Proceedings of The Seventh ACM Conference On Electronic Commerce*, 2006, pp. 228–23.

[33] D. Gruhl, D. Liben-Nowell, R. Guha, and A. Tomkins, "Information diffusion through blogspace," in *Proceedings of Thirteenth International World Wide Web Conference*, ser. WWW '04. New York, NY, USA: ACM, 2004, pp. 491–501.

[34] J. Leskovec, M. McGlohon, C. Faloutsos, N. Glance, and M. Hurst, "Cascading behavior in large blog graphs," in *Proceeding of the Seventh SIAM International Conference on Data Mining*, 2007.

[35] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan, "Group formation in large social networks: Membership, growth, and evolution," in *Proceedings of 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '06. New York, NY, USA: ACM, 2006, pp. 44–54.

[36] D. Newth, *Complex Science for a Complex World: Exploring Human Ecosystems with Agents*. Canberra: ANU E Press, 2006, ch. (5) The Structure of Social Networks.

[37] D. Acemoglu and A. Ozdaglar, "Graph theory and social networks," *Lecture material, Available at http://economics.mit.edu/files/4620*.

[38] W3C, "The platform for privacy preferences 1.0 (p3p1.0) specification," *Available at http://www.w3.org/TR/P3P/*, 2010.

[39] S. Milgram, "The small world problem," *Psychology Today Learning*, vol. 1, no. 1, pp. 60–67, 1967.

[40] Eight properties of social network (1-4). [Online]. Available: http://oddlee.blogspot.com/2007/09/eight-properties-of-social-network-14.html

[41] (2011) Linkedin press center. [Online]. Available: http://press.linkedin.com/

[42] Clustering coefficient. [Online]. Available: http://en.wikipedia.org/wiki/Clustering_coefficient

[43] R. Kumar, J. Novak, and A. Tomkins, "Structure and evolution of on-line social networks," *In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 611–617, 2006.

[44] S. Golder, D. Wilkinson, and B. Huberman, “Rhythms of social interaction: messaging within a massive on-line network,” *In: Proceedings of 3rd International Conference on Communities and Technologies*, 2007.

[45] A. Java, X. Song, T. Finin, and B. Tseng, “Why we twitter: understanding microblogging usage and communities,” *In: Proceedings of the Joint 9th WEBKDD and 1st SNA-KDD Workshop*, pp. 56–65, 2007.

[46] Stumbleupon. [Online]. Available: http://en.wikipedia.org/wiki/Stumbleupon

[47] D. J. Watts, *Small worlds: the dynamics of networks between order and randomness*. Princeton University Press, 1999.

[48] A. Clauset, M. E. J. Newman, and C. Moore, “Finding community structure in very large networks,” *Physical Review E*, vol. 70, no. 6, p. 066111, 2004.

[49] F. Cena, A. Dattolo, E. W. D. Luca, P. Lops, T. Plumbaum, and J. Vassileva, “Semantic adaptive social web,” in *UMAP Workshops*, ser. Lecture Notes in Computer Science, L. Ardissono and T. Kuflik, Eds., vol. 7138. Springer, 2011, pp. 176–180.

[50] G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions,” *IEEE Transaction on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734–749, 2005.

[51] T. H. Haveliwala, A. Gionis, and P. Indyk, “Scalable Techniques for Clustering the Web,” in *Proc. of the WebDB Workshop*, 2000, pp. 129–134.

[52] H. Marmanis and D. Babenko, *Algorithms of the Intelligent Web*, 1st ed. Manning Publications, June 2009.

[53] J. Leskovec, K. J. Lang, and M. Mahoney, “Empirical comparison of algorithms for network community detection,” in *Proceedings of the 19th inter-*

national conference on World wide web, ser. WWW '10. New York, NY, USA: ACM, 2010, pp. 631–640.

[54] C. A. R. Pinheiro, "Community detection to identify fraud events in telecommunications networks," in *SAS SUGI Proceedings: Customer Intelligence*. SAS Global Forum 2012, 2012. [Online]. Available: http://support.sas.com/resources/papers/proceedings12/106-2012.pdf

[55] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3-5, pp. 75 – 174, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/B6TVP-4XPYXF1-1/2/99061fac6435db4343b2374d26e64ac1

[56] P. Berkhin, "Survey Of Clustering Data Mining Techniques," Accrue Software, San Jose, CA, Tech. Rep., 2002.

[57] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data Clustering: A Review," *ACM Computing Surveys (CSUR)*, vol. 31, no. 3, pp. 264–323, 1999.

[58] R. Xu and I. Wunsch, "Survey of clustering algorithms," vol. 16, no. 3, pp. 645–678, May 2005.

[59] J. C. Gower, "A General Coefficient of Similarity and Some of Its Properties," *Biometrics*, vol. 27, no. 4, pp. 857–871, 1971.

[60] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988.

[61] H. Mocian, "Survey of distributed clustering techniques," Imperial College London, MSc Internal Research Project, 2009.

[62] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*, us ed ed. Addison Wesley, May 2005.

[63] N. Mishra, R. Schreiber, I. Stanton, and R. Tarjan, "Clustering Social Networks," in *Algorithms and Models for the Web-Graph*, ser. Lecture Notes in

Computer Science, A. Bonato and F. R. K. Chung, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, vol. 4863, ch. 5, pp. 56–67.

[64] Discovering who to follow. [Online]. Available: http://blog.twitter.com/2010/07/discovering-who-to-follow.html

[65] K. M. Hammouda, "Web Mining: Clustering Web Documents A Preliminary Review," University of Waterloo, Tech. Rep., 2001.

[66] B. Stein and M. Busch, "Density-based cluster algorithms in low dimensional and high-dimensional application," in *Second International Workshop On Text-Based Information Retrieval (TIR 05)*, 2005.

[67] E. R. Hruschka, R. J. G. B. Campello, A. A. Freitas, and P. Leon, "A Survey of Evolutionary Algorithms for Clustering," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 39, no. 2, pp. 133–155, March 2009.

[68] R. Xu and D. Wunsch, *Clustering (IEEE Press Series on Computational Intelligence)*, illustrated edition ed. Wiley-IEEE Press, October 2008.

[69] M. Naughton, N. Kushmerick, and J. Carthy, "Clustering sentences for discovering events in news articles," in *Advances in Information Retrieval*, ser. Lecture Notes in Computer Science, M. Lalmas, A. MacFarlane, S. Rüger, A. Tombros, T. Tsikrika, and A. Yavlinsky, Eds. Springer Berlin Heidelberg, 2006, vol. 3936, pp. 535–538.

[70] B. S. Everitt, S. Landau, and M. Leese, *Cluster Analysis*, 4th ed. Wiley, January 2009.

[71] B. Y. Wu and K.-M. Chao, *Spanning Trees and Optimization Problems*. Chapman and Hall/CRC Press, 2004.

[72] T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Wu, "An efficient k-means clustering algorithm: analysis and implementation,"

Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 24, no. 7, pp. 881–892, Jul 2002.

[73] S. B. Kotsiantis and P. E. Pintelas, “Recent advances in clustering: A brief survey,” *WSEAS Transactions on Information Science and Applications*, vol. 1, pp. 73–81, 2004.

[74] D. Arthur and S. Vassilvitskii, “k-means++: the advantages of careful seeding,” in *SODA ’07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[75] S. G. Rajeev, R. Rastogi, and K. Shim, “Rock: A robust clustering algorithm for categorical attributes,” in *Information Systems*, 1999, pp. 512–521.

[76] “Dbscan,” Wikipedia, the free encyclopedia, November 2010. [Online]. Available: http://en.wikipedia.org/wiki/DBSCAN

[77] C. C. Aggarwal, “Towards meaningful high-dimensional nearest neighbor search by human-computer interaction,” in *In ICDE*, 2002, pp. 593–604.

[78] “Delicious.com,” Wikipedia, the free encyclopedia, September 2010. [Online]. Available: http://en.wikipedia.org/wiki/Delicious.com

[79] M. E. J. Newman and M. Girvan, “Finding and evaluating community structure in networks,” *Physical Review E*, vol. 69, no. 2, p. 026113, 2004.

[80] M. E. J. Newman, “Fast algorithm for detecting community structure in networks,” *Physical Review E*, vol. 69, no. 6, p. 066133, 2004.

[81] B. H. Good, Y.-A. de Montjoye, and A. Clauset, “Performance of modularity maximization in practical contexts,” *Physical Review E*, vol. 81, no. 4, p. 046106, 2010.

[82] L. Danon, A. D. Guilera, J. Duch, and A. Arenas, “Comparing community structure identification,” *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2005, no. 9, pp. P09 008–09 008, 2005.

[83] C. Blundo, E. D. Cristofaro, and P. Gasti, “Espresso: Efficient privacy-preserving evaluation of sample set similarity,” *CoRR*, vol. abs/1111.5062, 2011.

[84] A. Lancichinetti, S. Fortunato, and F. Radicchi, “Benchmark graphs for testing community detection algorithms,” *Physical Review E*, vol. 78, no. 4, p. 046110, 2008. [Online]. Available: http://sites.google.com/site/andrealancichinetti/files

[85] M. Cha, A. Mislove, and K. P. Gummadi, “A measurement-driven analysis of information propagation in the flickr social network,” *In Proceedings of the 18th International World Wide Web Conference (WWW’09)*, 2009. [Online]. Available: http://socialnetworks.mpi-sws.org/data-www2009.html

[86] M. Girvan and M. E. J. Newman, “Community structure in social and biological networks,” *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, June 2002.

[87] G. K. Orman, V. Labatut, and H. Cherifi, “Qualitative comparison of community detection algorithms.” in *DICTAP (2)*, ser. Communications in Computer and Information Science, H. Cherifi, J. M. Zain, and E. El-Qawasmeh, Eds., vol. 167. Springer, 2011, pp. 265–279.

[88] F. Moradi, T. Olovsson, and P. Tsigas, “An evaluation of community detection algorithms on large-scale email traffic.” in *SEA*, ser. Lecture Notes in Computer Science, R. Klasing, Ed., vol. 7276. Springer, 2012, pp. 283–294.

[89] G. K. Orman, V. Labatut, and H. Cherifi, “Comparative evaluation of community detection algorithms: A topological approach,” *CoRR*, vol. abs/1206.4987, 2012.

[90] S. Papadopoulos, Y. Kompatsiaris, A. Vakali, and P. Spyridonos, “Community detection in social media,” *Data Mining and Knowledge Discovery*, vol. 24, pp. 515–554, 2012. [Online]. Available: http://dx.doi.org/10.1007/s10618-011-0224-z

[91] A. Lancichinetti and S. Fortunato, “Community detection algorithms: A comparative analysis,” *Physical Review E*, vol. 80, no. 5, p. 056117, Nov 2009. [Online]. Available: http://link.aps.org/doi/10.1103/PhysRevE.80.056117

[92] S. Fortunato, V. Latora, and M. Marchiori, “Method to find community structures based on information centrality,” *Physical Review E*, vol. 70, no. 5, p. 056104, 2004.

[93] S. Fortunato and M. Barthélemy, “Resolution limit in community detection,” *The National Academy of Sciences of the USA*, vol. 104, no. 1, pp. 36–41, 2007.

[94] Z. Li, S. Zhang, R.-S. Wang, X.-S. Zhang, and L. Chen, “Quantitative function for community detection,” *Physical Review E*, vol. 77, no. 3, p. 036109, 2008.

[95] A. Arenas, A. Fernández, and S. Gómez, “Analysis of the structure of complex networks at different resolution levels,” *New Journal of Physics*, vol. 10, p. 053039, 2008.

[96] A. Lancichinetti and S. Fortunato, “Limits of modularity maximization in community detection,” *Physical Review E*, vol. 84, no. 6, p. 066122, Dec 2011. [Online]. Available: http://link.aps.org/doi/10.1103/PhysRevE.84.066122

[97] A. Khadivi, A. A. Rad, and M. Hasler, “Community detection enhancement in networks using proper weighting and partial synchronization,” *Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 3777–3780, 2010.

[98] A. Khadivi, A. A. Rad, M. Hasler, “Network community-detection enhancement by proper weighting,” *Physical Review E*, vol. 83, no. 4, p. 046104, 2011.

[99] J. W. Berry, B. Hendrickson, R. A. LaViolette, and C. A. Phillips, “Tolerating the community detection resolution limit with edge weighting,” *Proceedings of The National Academy of Sciences of the USA*, vol. 83, no. 5, p. 056119, 2011.

[100] B. Yan and S. Gregory, “Detecting community structure in networks using edge prediction methods,” *CoRR*, vol. abs/1201.3466, 2012.

[101] P. D. Meo, E. Ferrara, G. Fiumara, and A. Provetti, “Enhancing community detection using a network weighting strategy,” *Information Sciences*, vol. 222, pp. 648–668, 2013. [Online]. Available: Available online at: http://www.sciencedirect.com/science/article/pii/S0020025512005488

[102] Y. Kang and S. Choi, “Common neighborhood sub-graph density as a similarity measure for community detection,” *Proceedings of the 16th International Conference on Neural Information Processing ICONIP 09*, vol. 1, pp. 175–184, 2009.

[103] A. Acar and Y. Muraki, “Twitter for crisis communication: lessons learned from japan’s tsunami disaster,” *Int. J. Web Based Communities*, vol. 7, no. 3, pp. 392–402, July 2011.

[104] S. Dumenco. (2011, April) A very brief (cartoon) history of social influence. [Online]. Available: http://bit.ly/hmXCif

[105] B. Solis. (2011, February) The interest graph on twitter is alive: Studying starbucks top followers. [Online]. Available: http://bit.ly/geE0i4

[106] J. Scott, *Social Network Analysis: a Handbook*. SAGE, 2000.

[107] R. A. Hanneman and M. Riddle, *Introduction to Social Network Methods*. Department of Sociology at the University of California, Riverside, 2005, ch. (10) Centrality and Power.

[108] D. Crandall, D. Cosley, D. Huttenlocher, J. Kleinberg, and S. Suri, "Feedback effects between similarity and social influence in on-line communities," *In: Proceedings of the 14th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, pp. 160–168, 2008.

[109] Wikipedia. (2011) Social psychology. [Online]. Available: http://en.wikipedia.org/wiki/Social_psychology

[110] N. Triplett, "The dynamogenic factors in pacemaking and competition," *The American Journal of Psychology*, vol. 9, no. 4, pp. 507–533, 1898.

[111] L. Festinger, *A Theory of Cognitive Dissonance*. Stanford University Press, 1957.

[112] L. Rashotte, "Social influence," *Blackwell Encyclopedia of Sociology*, pp. 4426– 4429, 2007.

[113] J. Tang, J. Sun, C. Wang, and Z. Yang, "Social influence analysis in large-scale networks," *In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '09)*, pp. 807–816, 2009.

[114] P. Lazarsfeld and R. K. Merton, "Friendship as a social process: A substantive and methodological analysis," *In Freedom and Control in Modern Society*, pp. 18–66, 1954.

[115] M. McPherson, L. Smith-Lovin1, and J. M. Cook, “Birds of a feather: Homophily in social networks,” *Annual Review of Sociology*, vol. 27, pp. 415–444, 2001.

[116] H. P. Young, “The diffusion of innovations in social networks,” *In The Economy as a Complex Evolving System*, vol. 3, 2003.

[117] A. Anagnostopoulos, R. Kumar, and M. Mahdian, “Influence and correlation in social networks,” *In Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD ’08)*, pp. 7–15, 2008.

[118] M. Granovetter, “The strength of weak ties,” *American Journal of Sociology*, vol. 78, no. 6, pp. 1360–1380, 1973.

[119] J. Sun and J. Tang, *Social Network Data Analysis*. Springer Science+Business Media, 2011, ch. (7) A Survey of Models and Algorithms for Social Infuence Analysis.

[120] M. Granovetter, “Economic action and social structure: The problem of embeddedness,” *American Journal of Sociology*, vol. 91, no. 3, pp. 481–510, 1985.

[121] C. Marlow, M. Naaman, D. Boyd, and M. Davis, “Ht06, tagging paper, taxonomy, flickr, academic article, toread,” *In Proceeding of the seventeenth conference on Hypertext and hypermedia HYPERTEXT ’06*, pp. 31–40, 2006.

[122] X. Shi, J. Zhu, R. Cai, and L. Zhang, “User grouping behavior in online forums,” *In Proc. 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 777–786, 2009.

[123] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan, “Learning influence probabilities in social networks,” *In Proceedings of the third ACM international conference on Web search and data mining, WSDM ’10*, pp. 241–250, 2010.

[124] G. Kossinets and D. J. Watts, "Empirical analysis of an evolving social network," *Science*, vol. 311, no. 5757, pp. 88–90, 2006.

[125] M. Richardson and P. Domingos, "Mining knowledge-sharing sites for viral marketing," *In Proc. of the Eighth ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD'02)*, pp. 61–70, 2002.

[126] M. Granovetter, "Threshold models of collective behavior," *The American Journal of Sociology*, vol. 83, no. 6, pp. 1420–1443, 1978.

[127] K. Saito, R. Nakano, and M. Kimura, "Prediction of information diffusion probabilities for independent cascade model," *In Proceedings of the 12th International Conference on Knowledge-Based Intelligent Information and Engineering Systems, Part III (KES'08)*, 2008.

[128] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions," *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.

[129] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. S. Glance, "Cost-effective outbreak detection in networks," *In Proc. of the 13th ACM Int. Conf. on Knowledge Discovery and Data Mining (KDD'07)*, pp. 420–429, 2007.

[130] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," *In Proc. of the 15th ACM Int. Conf. on Knowledge Discovery and Data Mining (KDD'09)*, pp. 199–208, 2009.

[131] I. Cantador, P. Brusilovsky, and T. Kuflik, "2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011)," in *Proceedings of the 5th ACM conference on Recommender systems*, ser. RecSys 2011. New York, NY, USA: ACM, 2011.

[132] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338–353, 1965.

[133] M. Hellmann, “Fuzzy logic introduction,” Retrived from http://epsilon.nought.de/tutorials/fuzzy/fuzzy.pdf, Université de Rennes, France, 2001.

[134] L. A. Zadeh, “Making computers think like people,” *IEEE. Spectrum*, vol. 8, pp. 26–32, 1984.

[135] F. Baig, M. W. Ashraf, Z. Ahmed, M. Imran, S. Tayyaba, and M. S. Khan, “Design and simulation of fuzzy logic based elid grinding control system,” *International Journal of Advanced Technology and Engineering Research (IJATER)*, vol. 3, no. 1, pp. 79–88, 2013.

[136] S. M. Rahman and N. T. Ratrout, “Review of the fuzzy logic based approach in traffic signal control: Prospects in Saudi Arabia,” *Journal of Transportation Systems Engineering and Information Technology*, vol. 9, no. 5, pp. 58–70, 2009.

[137] R. Rojas, *Neural Networks: A Systematic Introduction.* New York, NY, USA: Springer-Verlag New York, Inc., 1996.

[138] Fuzzy logic: Example 7: Choosing a job. [Online]. Available: http://www.wolfram.com/products/applications/fuzzylogic/examples/job.html

[139] K. AlFalahi, Y. Atif, and S. Harous, “Community detection in social networks through similarity virtual networks,” in *Proceedings of the 4th Business Applications of Social Network Analysis Workshop, (BASNA 2013) in Conjunction With the IEEE and ACM, (ASONAM 2013).* ACM, 2013.

[140] K. AlFalahi, Y. Atif, and A. Abraham, “Models of influence in online social networks,” *International Journal of Intelligent Systems*, vol. 29, no. 2, pp. 161–183, 2014.

[141] C. Kahraman, “Multi-criteria decision making methods and fuzzy sets,” in *Fuzzy Multi-Criteria Decision Making*, ser. Springer Optimization and Its Applications, C. Kahraman, Ed. Springer US, 2008, vol. 16, pp. 1–18.

[142] V. Peneva and P. Ivan, “Multicriteria decision making based on fuzzy relations,” *Cybernetics and Information Technologies*, vol. 8, no. 4, pp. 3–12, 2008.

List of Publications

K. Al-Falahi, Y. Atif, "Community Aware Influence Maximization in Social Networks Using Fuzzy Logic", Computing Journal, Springer, special issue on Computational Intelligence, Submitted.

K. Al-Falahi, Y. Atif, A. Abraham, "Models of Influence in Online Social Networks", International Journal of Intelligent Systems, Vol. 29(2), pp. 161-183, Wiley. 2014.

K. Al-Falahi, Y. Atif, "Models of Influence in Online Social Networks", Information and Communication Technologies Research Forum, ICTRF'13, Abu Dhabi, UAE. 2013.

K. Al-Falahi, Y. Atif, S. Harous, "Community Detection In Social Networks through Similarity Virtual Networks", In Proceedings of the 4th Business Applications of Social Network Analysis Workshop, (BASNA 2013), in conjunction with the IEEE and ACM, (ASONAM 2013), Niagara Falls, Canada. 2013.

K. Al-Falahi, S. Harous, Y. Atif, "Clustering Algorithms For Intelligent Web", In International Journal of Computational Complexity and Intelligent Algorithms (L. Wang ed.), Inderscience Publishers. 2013.

K. Al-Falahi, N. Mavridis, Y. Atif, "Social Networks and Recommender Systems: A World of Current and Future Synergies", In Computational Social Networks: Tools, Perspectives and Applications (A. Abraham and A.-E. Hassanien, eds.), pp. 445-465. Springer. 2012.

K. Al-Falahi, S. Harous, Y. Atif, "A Comparative Study of Clustering Algorithms", In International Journal of Virtual Communities and Social Networking (S. Dasgupta ed.), Vol. 3(3), pp. 1-18, IGI Publishing. 2011.

K. Al-Falahi, Y. Atif, S. Elnaffar, "Social Networks: Challenges and New Opportunities", In Proceedings of the IEEE International Symposium on Social Computing and Networking, (SocialNet 2010), pp. 804-808, Hangzhou, China. 2010.