Spring 5-4-2021

Graph Mining and Its Applications in Online Social Networks

Xueting Liao

Recommended Citation
Liao, Xueting, "Graph Mining and Its Applications in Online Social Networks." Dissertation, Georgia State University, 2021.
doi: https://doi.org/10.57709/22565128

GRAPH MINING AND ITS APPLICATIONS IN ONLINE SOCIAL NETWORKS

by

Xueting Liao

Under the Direction of Yubao Wu, PhD and Xiaojun Cao, PhD

ABSTRACT

Online Social Networks (OSNs) have become prevalent in people's daily life. Facebook, Twitter, and Instagram are among the most popular social networking platforms. There are over 3.6 billion people using social networks worldwide in 2020. The number is still expected to grow with the development of portable smart devices. With a large number of users, the impact of online social networks increases incredibly. The social network is not only changing how we communicate and interact with each other in daily life, but also how people make decisions.

Social graphs or networks are often used to depict the personal relations of Internet users on these platforms. In a social network, the nodes could represent individuals in the network, and the edges could represent the social interactions between them. It is crucial to mine useful knowledge from the huge and complex social network graphs efficiently. One of the

fundamental problems in Online Social Network is measuring the similarity between nodes effectively and efficiently. It is the first step for many applications like querying and ranking, community detection and link prediction.

As a result, we propose a series of works for effectively retrieving meaningful information in Online Social Networks. Firstly, we investigate the similarity measurement and propose a similarity measurement model to help measure the proximity between nodes in a graph. The proposed approach is validated by real-world social networking data. Then, we investigate the credibility of social influencers and propose a framework for content and influencer trustworthiness in online social networks. The texts and images are jointly considered and properly balanced. The proposed framework works as an approach to deal with the challenges for trustworthiness in the current online social networking environment in the application level. Then, we investigate utilizing online social networks to gather open-source intelligence from the Coronavirus-related events. All the proposed solutions are thoroughly discussed and validated with extensive evaluations.

INDEX WORDS: Graph Mining, Similarity Measurement, Information Retrieval, Community, Online Social Networks.

GRAPH MINING AND ITS APPLICATIONS IN ONLINE SOCIAL NETWORKS

by

Xueting Liao

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

in the College of Arts and Sciences

Georgia State University

2021

GRAPH MINING AND ITS APPLICATIONS IN ONLINE SOCIAL NETWORKS

by

Xueting Liao

Committee Co-Chair: Yubao Wu

Xiaojun Cao

Committee: Yi Pan

Yanqing Zhang

Yichuan Zhao

Electronic Version Approved:

Office of Graduate Studies

College of Arts and Sciences

Georgia State University

May 2021

DEDICATION

I would like to show my gratitude to my family and all friends for their support and understanding, especially my advisor Dr. Xiaojun Cao and Dr. Yubao Wu for their comments that have greatly improved the manuscript. I cannot finish my Ph.D. without their encouragement.

ACKNOWLEDGMENTS

It would be impossible for me to finish the dissertation without the guidance of my committee members, help from my group, and support from my family and friends. I would like to show my great gratitude to all of them.

First and most important, I would like to give my deepest thanks to my advisors, Dr. Xiaojun Cao and Dr. Yubao Wu, for their generous guidance, encouragement, assistance, caring, patience, and support. They patiently inspired me and financially supported my research. Dr. Cao not only provided valuable suggestions and guidance when I was trapped in my research but also motivated me to pursue higher career growth. His hard-working, brilliant and keen has set a good example for me to follow. Dr. Wu has taught me not only the methodologies used for my study but also the right attitudes towards my research. I really appreciate their help.

It is very grateful and a great honor to have Dr. Yi Pan, Dr. Yanqing Zhang and Dr. Yichuan Zhao in my committee. They gave me great supports for my Ph.D. study and spent spare time participating in my defense committee despite their busy schedules. Their helpful and insightful advice provide me a broader view of the researches.

I am also very thankful to the professors and staffs in our department. Without their countless help, I could not live and study in Atlanta as easy and convenient as now. And I would not be able to pursue a Ph.D. degree without the Computer Science department and Molecular Basis of Disease's (MBD) generous funding.

Last but not least, it is a pleasure to thank everybody who made the dissertation possible, especially during this pandemic season, as well as expresses my apologies that I could not mention personally one by one.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

- OSN - Online Social Network
- DCG - Discounted Cumulative Gain
- NDCG - Normalized Discounted Cumulative Gain
- USPD - Undisclosed Sponsored Post Detection
- DSP - Disclosed Sponsored Post
- USP - Undisclosed Sponsored Post
- UP - Unsponsored Post
- RP - Random Predictor
- LR - Logistic Regression
- SVM - Support Vector Machine
- RF - Random Forest
- TGC-PDA - Tripartite Graph Clustering on Pandemic Data Analysis
- NMF - Non-negative Matrix Factorization

CHAPTER 1
INTRODUCTION

This chapter is an introduction to this dissertation. It focuses on the background and motivation of dissertation research including online social networks, network structure and graph mining. The structure of the dissertation is introduced at the end of this chapter.

1.1 Online Social Networks

Network is a large system that consists of many similar parts connecting together with components to communicate or move among the parts [2]. The parts can be street, computer, and people. If the parts are computers and their peripherals, a computer network is formed. The components that can be used to formulate a computer network are of microwaves stations, fiber optics and communication satellites etc.. A computer network is capable of sharing information, instructions and resources such as files and documents. Similarly, a social network is an abstract network connecting people or organizations. Comparing with a computer network that is made up with machines connected by a set of cables, a social network consists of a group of people (or organizations or other social entities) connected by a set of social relationships, such as friendship, citation or information interchange [3]. In daily life, it is common to see coupon books being shared among family and friends, in this case, such social network shares coupon books as the resource among people. In addition to entitative format, the booming electronic social networking platforms provide people with opportunities to communicate and interact online, which brings Online Social Networks to the world. Meanwhile, with the development of electronic products, it makes possible to

connect to anyone from anywhere, at any time on any device. More and more people get their news from social media, make their decisions based on social media and share their daily life through social media. With a large number of people using social network platforms, the impact of social networks increases incredibly. There are over 3.6 billion social network users in 2020, which accounts for more than 71 percent of total internet users [4].

Online Social Networks (OSNs) provide a large amount of data containing rich and valuable information, which can only be able to processed by machines. This situation motivates data mining researchers to work on OSNs to extract meaningful information. Such information could be utilized to make people's live better. For example, when a user goes to Linkedin's website, it will recommend people with similar professions to this user, which can largely help people connect with other professionals and explore new opportunities. However, if the data in OSNs has been maliciously leveraged by ill-intentioned people, the consequence will be severe and social utility will be decreased. For example, the leaked data from Facebook influenced people's decisions for the presidential election, which may have affected the 2016 US presidential election result [5]. Therefore, it is meaningful and valuable to analyze OSNs to provide people convenient or easeful environment and protect people from the effect of malicious or fake information. To analyze an OSN, the first step is to model the network as a mathematical structure for the subsequent methods.

Figure 1.1: An example of Online Social Network

1.2 Network Modeling

To represent an Online Social Network (OSN), the most popular data representation method is building a graph. A graph is a data structure with nodes and edges. The nodes can represent individuals, accounts or entities. The links or edges that connect the nodes can represent the interaction or relationship between nodes. Fig. 1.1 is an example of an online social network. The node is not limited to a user, it could also represent a device or an application. The edge can be an interaction, such as one user liking a tweet. It can also be a relationship. For example, in a graph where nodes represent users, we can connect two nodes if these two users are friends. The graph can be undirected (edges point both ways) or directed (each edge only points from one node to another, but not vice versa). For instance, the relationship between users in Facebook is undirected. Once one user adds the other user as his/her friend, they will become friends with each other. In contrast, the

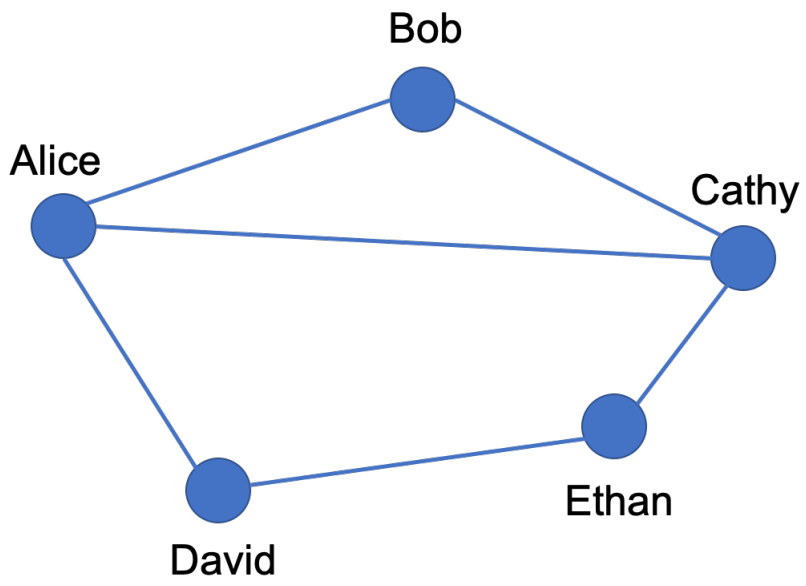

Figure 1.2: An example of the OSN with five users and friendship links

relationship between users in Twitter is directed. One user (the follower) follows another user (the followee) does not necessarily mean the followee would follow the follower. Therefore, when considering an OSN to model as a graph, the above mentioned situations need to be taken into account.

Fig.1.2 and Fig. 1.3 shows a graph representation of an OSN with five users. Fig.1.2 is purely based on the established friendship relationship. It includes a clique (a clique is a subset of nodes with every two different nodes adjacent in an undirected graph) formed by Alice, Bob and Cathy. Alice and Cathy are the people with the most friends in the OSN. However, if we use a different way to form a graph, for instance, people only sending more than ten messages over one month can build an edge, the graph representation may be as Fig. 1.3. This graph representation will be a directed graph, showing who send message to whom. And the weights on the edges indicate the number of messages sent. In this new

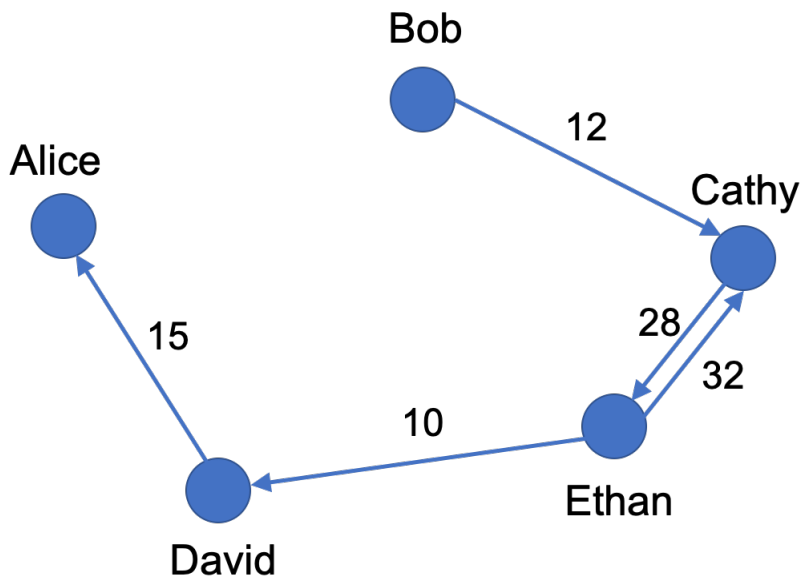

Figure 1.3: An example of the OSN which builds links with more than ten messages over one month

graph, Alice, Bob and Cathy are not in a clique any more. The graph cannot infer that Alice and Cathy have more friends than the other three. The large weight in two directions between Cathy and Ethan indicate they communicate more than other pairs. Thus, we can find these two types of representation of one OSN can lead to different analysis result, which also exhibit the rich modeling capabilities of graphs.

Analyzing Online Social Networks is a process of investigating social structure through the use of networks and graph theory. Different network structures can represent different information which can infer different knowledge. Therefore, finding a proper graph representation or network structure is a critical step for OSN analysis, which will also largely affect the subsequent selection of methods to mine knowledge.

1.3 Graph Mining

Data mining is a process of knowledge discovery in databases, which can be used to recognize patterns and extract information from the given data [6]. In contrast to data mining that targets mining information from isolated data, graph data mining or graph mining strives for better performance and innovation from structural information of the data. Knowledge discovery from interconnected and complex linked data that builds proper relations/interactions of the data raises a general problem for mining.

Graph mining has become increasingly important with broad applications such as text retrieval, web analysis, bioinformatics, computer vision, and social networks. For example, researchers use the graph representation to model the text in text retrieval, where nodes can represent paragraphs, sentences or words and edges can represent the relationship between two nodes, such as semantic and syntactic relationship. A directed graph can be adopted to represent the flow on the text, and undirected graph can be built if the order could be ignored in the co-occurrence terms analysis. If we need to consider the number of co-occurrences of terms in a document, we can also consider weighted edges, and use the number of co-occurrences to represent the weight for each edge. Then, with the sophisticated graph theory and graph mining methods, we can apply them to the rich modeling potential of graphs to solve the aforementioned text retrieval problem. Meanwhile, with the growing demand for analyzing large structured data, the graph mining has become a critical sub-domain in data mining. It can be used to analyze the properties of real-world graphs, predict the effect with the given structure and properties of a graph, and find subgraphs that is capable to compress

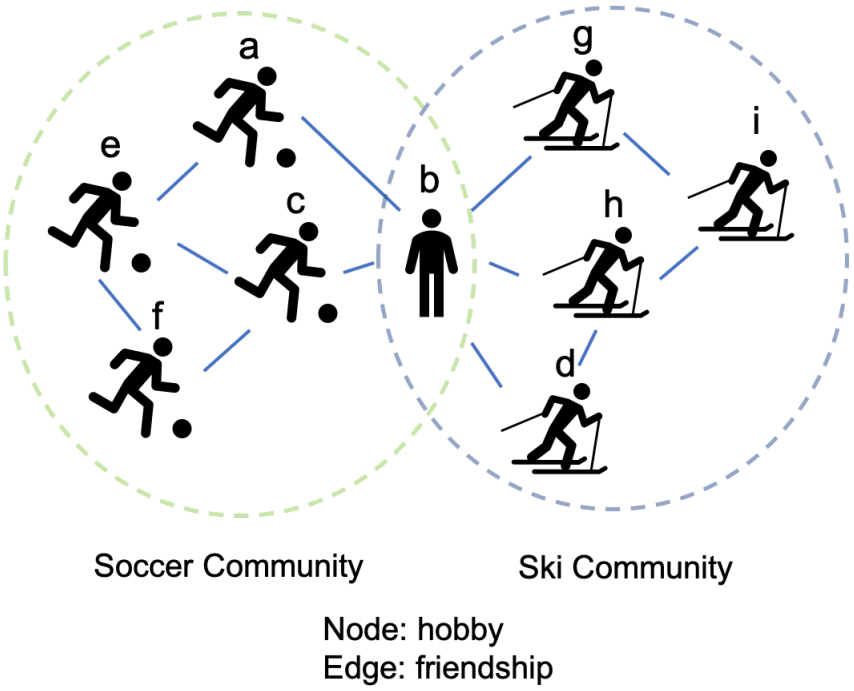

Figure 1.4: An example of the OSN with people having different hobbies in the Facebook platform

the data by abstracting instances of the substructures.

One critical category in graph mining is graph clustering. It is grouping the nodes of the graph into clusters by taking edge structure into consideration in such a way that nodes are densely connected within each group and sparsely connected between groups. These densely connected graphs are also called clusters. Graph clustering is based on the unsupervised learning technique in which the classes are unknown before being clustered. The clusters can be formed based on some similarities in the underlying graph-structured data. One straightforward example is finding the communities of Facebook users based on their exploring interests, as is shown in Fig. 1.4. In this example, a node represents the hobby of that user and an edge represents the friendship between two users. Person b is

the one interested in both soccer and ski. If we want to find communities in the figure, a straightforward idea is that people with same hobby should be more similar than people with different hobby. Thus, we can cluster users by their hobbies. Users in the green circle form a soccer community and users in the blue circle form a ski community. The formed communities are also overlapping communities, where user b is in both two communities simultaneously.

Query and ranking usually take place in a large complex graph with only local information known. The result will be a series of rated data of the query node. One common example is recommending most k relevant tweets when a user clicks one tweet (known as the query node). We can also take Fig. 1.4 as the other example, query is selecting a node, for instance, node a. Then, we rank the most 3 similar users to node a, and return the result as the application output. Link prediction is usually used to predict if there will be an edge between two nodes in the future, which can be utilized to recommend friends to a user in an online social media platform. For example in Fig. 1.4, we can predict if node f and node b will be friends in two months. More explanation about the detailed methods to solve aforementioned problems will be introduced in Chapter 3.

With different graph representations and different scenarios, different solutions could be provided. Therefore, when considering selecting proper methods from sophisticated graph theory and graph mining algorithms, suitable graph models needed to be carefully discussed. Meanwhile, the graph size can also affect the mining methods. When a graph is too large, parallel and distributed solutions would be needed. Thus, when analyzing the graph data,

the first step is to identify a scenario and build a graph model. Then, based on the graph size and requirement, we can come up with a solution for the tasks.

1.4 Dissertation Organization

The rest of the dissertation is organized as follows: Chapter 2 provides the network model definition. Chapter 3 analyzes the existing literature in the context of graph mining in online social networks. Chapter 4 discusses the problems and proposed method for similarity measurement in online social networks, which builds a theoretical foundation for our dissertation. Chapter 5 investigates the problem of credibility issues in online social networks, and provides the statistics of the real-world social network, which explain the usage of graph mining in an application aspect. Besides the unipartite graph, chapter 6 introduces the background of the tripartite graph in representing social networks and our approach to provide analysis for the current crisis in coronavirus pandemic. Chapter 7 discusses the limitations and potential directions for future work. Chapter 8 concludes our work.

CHAPTER 2
ONLINE SOCIAL NETWORK MODEL

The concept of online social networks, where relationships among online entities are represented as links in a network, has increasingly attracted attention in the past decades with the booming of social medias. Network refers to a real system such as a web network, a social network. The online social network model has the characteristics of social networks, which can be utilized for further research. The nodes and edges can have various meanings. In this chapter, we introduce the characteristics of social networks and the models that represent online social networks.

2.1 Online Social Network Characteristics

Usually, social networks are dynamic, the nodes and edges are added or removed over time. In general, social networks tend to show belowing characteristics [6]:

- Densification power law: Researchers used to think when a network evolves, the sum of degrees (i.e., number of edges incident to a node) expands linearly in the number of nodes. This is called constant average degree assumption. Nevertheless, the extensive experiments have shown that networks become more and more dense over time with the average degree growing instead. Therefore, the number of edges increases superlinerly as of the number of nodes. The densification follows the densification power law (or growth power law), as is shown below:

$$e(t) \propto n(t)^a \tag{2.1}$$

where $e(t)$ represents the number of edges and $n(t)$ represents the nodes of the graph at a timestamp t. The exponent a is between 1 and 2 strictly. If $a = 1$, it means constant average degree over time. If $a = 2$, it corresponds to an extremely dense graph where each node has edges to a constant fraction of all nodes.

- Shrinking diameter: The network diameter is the maximum distance between pairs of nodes. Empirical results have shown that the effective diameter has tendency to decrease as the network increases. This contradicts the early understanding that the diameter slowly increases with the network size increasing. Take the citation network as an example, the nodes are papers and an edge is a citation from one paper to another. Obviously, the edge is directed since the citation has direction and cannot be reversed. When a new node join the graph, it would have connections with previously existed nodes in the graph. It can be treated as building bridges to previous nodes, which will decrease the distances between pairs of nodes.

- Heavy-tailed out-degree and in-degree distributions: By observing the power law, the number of out-degrees of a node likely follows a heavy-tailed distribution. In power law, we have $1/n^a$, where n is the rank of the node in the order of decreasing out-degrees and a is between $(0, 2)$. The smaller value of a, the heavier the tail. The in-degrees also follow a heavy-tailed distribution, and it tends to be more skewed than the out-degrees distribution.

2.2 Social Network Models

In previous section, we described the characteristics and functionalities of Online Social Networks. We know that a social network is made up with people, organizations or other social entities, such as tweets and posts. They are nodes of the graph as for mathematical abstraction. The interaction or relationship between each node are edges of the graph. A node can consist of various attributes according to the social network. For example, a user in Instagram can contain following information:

- Personal Information: Gender, Location, Working information, Description about itself, Number of followers and followings.
- Historical posts: Text, Time, Images and Comments
- Profile photo

However, in other social network such as Linkedin, the attributes for a user is almost completely different from Instagram. A profile of a user may look like:

- Current position (Job title such as Professor, Software Engineer)
- Educational Background
- Working History
- Skills, Endorsements and Accomplishments

In addition, an edge can also provide different meanings according to social networks. For instance, in Facebook, a user can be a family member of another user, they are more

closely connected comparing to the colleagues. This situation could imply that the features of an edge such as weight and directionality are important and need to be considered.

To summarize, we can find that from the point of view of data mining, a social network is a heterogeneous and multirelational data set represented by a graph. The graph is usually huge. And both nodes and edges have attributes. To abstract social network and benefit our problem formulation, let us notate a social network graph $G = (V, E)$ where V represents the node set and E represents the edge set between nodes, where $V = \{v_1, v_2, ..., v_n\}$ and the total number of nodes $|V| = n$ and $E = \{e_1, e_2, ..., e_m\}$ and the total number of edges $|E| = m$.

CHAPTER 3
GRAPH MINING IN SOCIAL NETWORKS

Social network analysis, from a data mining perspective, could be called link analysis or graph mining as well. It includes analysis of the data, data management and storage, complexity considerations, visualizations and streaming data updating. As different graph representation can lead to different mining result, in this chapter, we firstly introduce the representations of graphs, then concepts and techniques in graph mining.

3.1 Graph Types

Usually, we use a set of nodes with edges connecting some of them as the definition of a graph. However, there could be several variations with different names. A graph can be a unipartite or a k-partite graph depending on whether the set of nodes pointed to by edges is the same as the set of nodes pointed from, or not. A k-partite graph can be partitioned into k different disjoint sets, with no edges connecting any two nodes inside each node set. Bipartite graph is a special case of k-partite/multipartite graph with two different independent sets. A tripartite graph is a k-partite graph with k equals 3. Besides, edges can have different weights, or not, which differentiates weighted graphs from unweighted graphs. Weighted graphs can be both unipartite or bipartite graphs, and unweighted graphs are a special case of weighted graphs with all weights being equal. A complete graph is a simple undirected graph with every pair of distinct nodes connecting with a unique edge. A complete digraph is a directed graph that a pair of unique edges, with one in each direction, connects every pair of distinct nodes. All of these special terms are formally defined as follows:

Definition 1 (Directed and undirected graph)**.** *In a directed graph $G = (V, E)$, each edge $(i, j) \in \varepsilon$ points from node i to node j. An undirected graph is a directed graph with edges pointing both directions, that is, $(i, j) \in \varepsilon \Rightarrow (j, i) \in \varepsilon$.*

Definition 2 (Weighted and unweighted graph)**.** *In a weighted graph $G = (V, E, W)$, where V represents the set of nodes, E represents the set of edges and W represents the corresponding weights of the edges. Unweighted graph is a special case of weighted graph with all weights equals to 1.*

Definition 3 (Unipartite and k-partite graph)**.** *In a k-partite graph, the set of nodes V consists of k disjoint sets of nodes: $V = V_1 \cup V_2 \cup ... \cup V_k$. Any edge connecting nodes with different sets, that is, $(i, j) \in E \Rightarrow$ if $i \in V_1$ then $j \notin V_1$. In other words, there are no edges between nodes of the same partition.*

Every graph $G = (V, E)$ can be represented by its adjacency matrix $\mathbf{A}$. The size of matrix $\mathbf{A}$ is $|V| \times |V|$, where the rows and columns represent the nodes of the graph and the entries indicate the existence or weights of the edges.

Definition 4 (Adjacency Matrix)**.** *The adjacency matrix $\mathbf{A}$ of a graph $G = (V, E)$ is a $|V| \times |V|$ matrix, such that*

$$\mathbf{A}_{i,j} = \begin{cases} w_{i,j} & \text{if } (i,j) \in E, \forall i,j \in 1,...,|V| \\ 0 & \text{otherwise} \end{cases}$$

The above definition is quite general and can be used for weighted and unweighted graphs. For the formal case, each value $w_{i,j}$ represents the weight between the edge (i, j), and for unweighted graphs, the weight of each edge is equal to one (i.e. $w_{i,j} = 1, \forall (i, j) \in E$).

With easy observation, we can find the adjacency matrix $\mathbf{A}$ is symmetric when the graph is undirected, with $\mathbf{A} = \mathbf{A}^\top$. However, for a directed graph, the previous equation cannot stand.

Following are some other important terminologies that are needed in graph mining:

- Node degree: number of neighbors of a node in a undirected graph
- In-degree: number of inbound links or in-edges in a directed graph
- Out-degree: number of outbound links or out-edges in a directed graph
- Adjacent node a node u is adjacent to a node v if there is an edge between u and v, i.e., $(u, v) \in E$
- Path: a sequence of adjacent nodes in a graph, and the length of a path equals to the number of edges
- Diameter of a graph: length of the longest shortest path

The various types of graphs make graph mining challenging, as different characteristics require different technologies to handle them. For example, a directed graph is characterized by asymmetrical matrices, such as adjacency matrix, which means spectral analysis will be more complex compared with undirected graphs.

3.2 Graph Mining Primitives

A graph is a ubiquitous data structure that can be utilized to model many real-world problems, especially in the social science domain, it is an abstract mathematical model of a

network. There are several primitive problems in analyzing the graph data, such as community detection, query and ranking, link prediction. Communities or clusters are usually groups of nodes highly connected or quite similar to each other than to members of other groups. Community detection aims to detect all communities in the entire graph. Query and ranking mean given a query node, the methods aim to rank all other nodes based on some similarity measurement. Link prediction is to predict whether there will be links between two nodes based on the existing graph information.

In recent years, the sizes of social networks have been exploding. Thus, it is crucial to mine useful knowledge from the huge and complex social network graphs efficiently. Besides, for non-industry groups, it is impossible for them to access the entire network. For instance, for the Instagram platform, it only provides a regular user a limited size graph with limited query functions, which means we can only get partial network structure in Instagram. Meanwhile, in many applications, we may only need to find the top-k nodes which are most similar to the query node, which provides us a new thinking process.

3.2.1 Community Detection Problem

Community detection in the graph is a popular research topic in graph mining. However, it is an ill-defined problem, which means there is no universal definition of the objects that one should be looking for [7]. Thus, there is no clear rules about how to assess the performance of different algorithms and how to compare them with each other. On the other hand, such ambiguity provides us lots of freedom to propose diverse approaches to the problem. The commonly used basic assumptions of the communities are that the nodes are densely con-

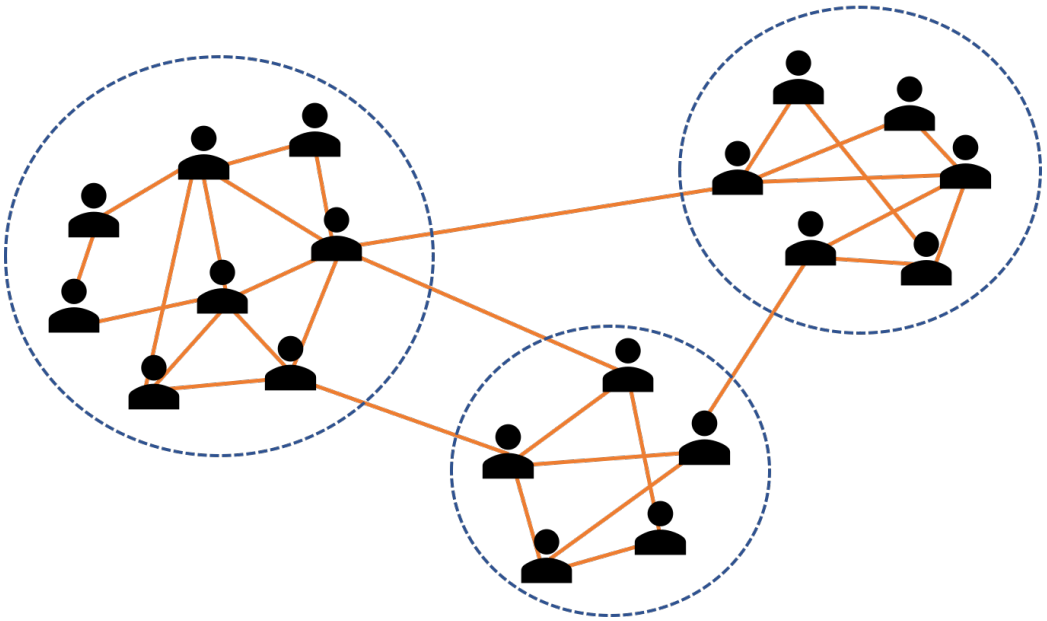

Figure 3.1: An example of community detection problem

nected within the community and sparsely connected to the nodes outside of the community. It is also known as graph partition [8], which partitions the whole graph into some disjoint communities. The algorithm developed by Newman is one of the most important on in this area [8]. However, it depends on good modularity measures and cannot work for an overlapping community. There are other types of approaches to detect communities including stochastic methods. This type of approach uses the concept of structural equivalence. It determines the group of nodes that connect to nodes in other communities in an equivalent way. This means a bipartite graph can be applied with these methods. In [9], the authors utilize Bayesian model to find communities in email networks, with topics of discussion and social links information. This nonlinear model will require more computational resources and not suitable for overlapping communities. In Fig. 3.1, by using a partition-based clus-

tering algorithm such as spectral clustering algorithm [10], the graph will be partitioned into three disjoint groups, as is shown in the dashed circles. We can find this graph has three communities, the nodes in the same community are highly connected comparing to the nodes in different communities. Community detection can provide us insights into the formation of the real graph and can be further applied to many critical commercial or non-business activities such as recommendations and link prediction.

To evaluate community detection, a metric called modularity [11] is introduced. It is calculated by below equation:

$$Q = \frac{1}{2m} \sum_{i,j} (A_{i,j} - \frac{k_i k_j}{2m}) \delta(g_i, g_j) \tag{3.1}$$

Where Q is the modularity, m is the total weight of all edges in the graph, $A_{i,j}$ is the edge weight between node i and node j. k_i is the total weight of all edges connecting node i and all other nodes in the graph. The function $\delta(g_i, g_j)$ evaluates if node i and node j belong to same community when it equals to one, otherwise zero. Modularity is used to decide the partition of the graph, and the graph that is not partitioned or those that place every node in its own group will have zero modularity value. Thus, we can use modularity to measure the community detection effectiveness, or we can even use modularity to do community detection.

3.2.2 Query and Ranking

Query and ranking is a task that usually takes place in a complex network and appears in information retrieval domain. By filtering and rating data, ranking strategies can provide a

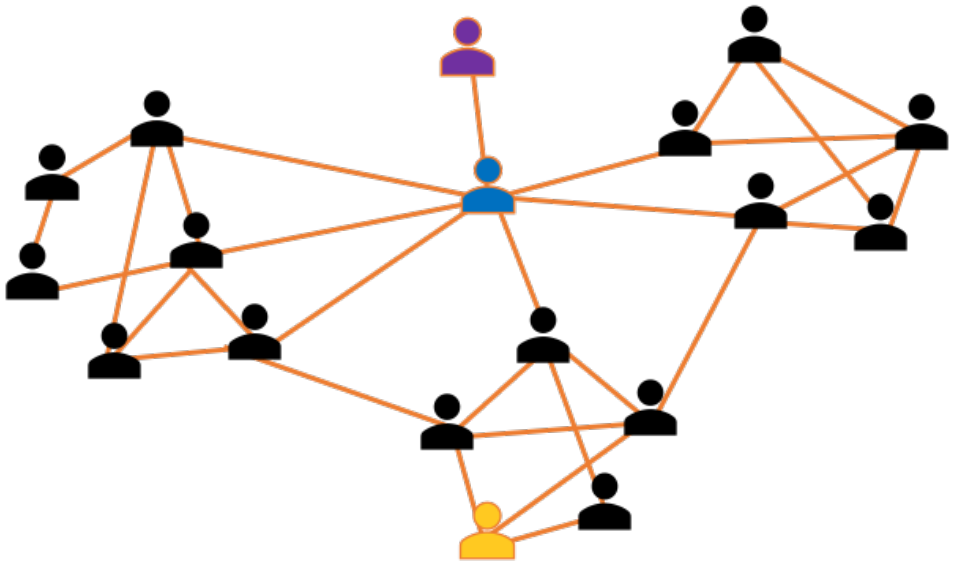

Figure 3.2: An example of finding top-k proximity query problem

small size of related data when exploring large amount of information. There are many ranking algorithms that follow different approaches focusing on various aspects of the problem, with various strategies that could be applied. Proximity measurement on graphs is one of the foundation topics in numerous graph-based applications. With different criteria, various proximity measurements have been proposed. Random walk based proximity measurements have been demonstrated to outperform many other proximity measures, such as those based on common neighbors, shortest paths, or network flows. The random walk based proximity measure captures well on the community structure in a network and it can be computed efficiently. For example, in Fig. 3.2, we mark query person blue and we want to rank the yellow and purple node with their proximity with the query person. If we use hop number as the proximity measurement, then, the purple person is closer to the blue query person than yellow person since it is adjacent to the query person. However, if we use the number

of paths to the query node, the result will be reversed. Yellow person is closer to the query person since it has four paths connecting to the query person. There are numerous random walk based proximity measure including aforementioned disciplines, PageRank [12], random walk with restart [13] and SimRank [14].

With the proximity measure provided, the application of query and ranking can be solved, which is one of the fundamental problem in data mining domain. Ranking problem is given a query node q and a collection of nodes D, sort or rank the nodes in D according to some disciplines so that the "best" results appears in front of the result list. In lots of real-world applications, it is not necessary to compute the all pairwise proximity no matter in time complexity aspect or application scenarios aspect. Thus, the top-k proximity query problem is defined, which is shown below:

Definition 5. *Given a undirected graph $G(V, E)$, a query node q and a number k. Let $\mathbf{p}_i$ represent the proximity value of node i with regard to node q. The top-k proximity query problem is to find a node set $K \subseteq V \neg \{q\}$ such that $|K| = k$ and $\mathbf{p}_i > \mathbf{p}_j$ for any node $i \in K$ and $j \in V \neg K \neg \{q\}$.*

There are many research working on effectively retrieve the top-k nodes with regards to the query node. In this work, we will use top-k proximity query problem as an application to measure the effectiveness of the proximity measurement proposed. Specifically, we want to sort the friends of a random user, then return the top-k relevant friends in Online Social Networks. The discounted cumulative gain (DCG) measures the quality of the results in a ranked list, where items in that list are ranked by the scale of one to five. It checks the

relevance for each randomly selected user or node. Normalized DCG (NDCG) is defined to calculate this measure across the independent queries, the result lists for which might all be of different length. NDCG is computed as:

$$\mathrm{NDCG}_p = \frac{DCG_p}{IDCG_p}$$

If more than one user is selected, the NDCG values are averaged to obtain an accuracy value, whereas p is the number of chosen users or nodes. This could provide us more insights about the proposed measurement.

3.2.3 Link Prediction

Link prediction attracts increasing attentions in computer science domain. It could be used for identifying spurious interactions, extracting missing information, and predicting future links in the evolving networks. For instance, the potential links could be recommended as promising friendship in online social networks. There are lots of similarity-based algorithms that are developed for link prediction. The main idea is basing on the similarities between a pair of nodes, the link connecting more similar nodes are more likely to be the potential link. The empirical evidence also shows that entities are more likely to interact with each other if they are similar. Node similarity can be defined by the attributes of the nodes or by its structure. The later one is only based on the network topology. The structural similarity measures can be categorized into local [15] and global [16] methods.

Local similarity can be measured by common neighbors, Jaccard Index [17]. Common neighbors is the simplest local technique. It is defined by the number of shared neighbors

between two nodes [18]. Despite its simplicity, this measure performs pretty well on most real-world networks and beats many complex approaches. It is also the basis for other approaches presented later. Using this method to compute similarity for all possible pairs results in a local link prediction technique with $O(vk^3)$ time complexity, where v represents number of nodes and k represents the maximum degree of a node. Jaccard Index measures the ratio of shared neighbors in the complete set of neighbors for two nodes. It is a variation of the common neighbors method with penalization for non-shared neighbors. The time complexity is $O(vk^3)$.

Global similarity can be calculated by Katz Index [19] and random walk based methods such as random walk with restart [13] and SimRank [14]. The Katz Index sums the influence of all possible paths between two pairs of nodes, with increasing penalization for path length. The mathematical expression is:

$$s_{xy} = \beta A_{xy} + \beta^2 (A_{xy})^2 + \beta^3 (A_{xy})^3 ... \tag{3.2}$$

where β is a parameter controlling the path weights. A small β could make it close to common neighbor because long paths would have less contribution. The matrix form is defined as:

$$S = (I - \beta A)^{-1} - I \tag{3.3}$$

where $[S]_{i,j}$ is the similarity score between node i and node j. The Katz index has a great predictive power but the high algorithmic complexity due to the matrix inverse operation. The time complexity of this method is $O(vk + v^3 + v)$, where the $O(vk)$ is for matrix

subtraction and scalar multiplication, $O(v^3)$ is for matrix inversion, and $O(v)$ comes from the subtraction of the diagonal elements in the identity matrix. Thus, the time complexity is $O(v^3)$. For random walk based methods, they are using the Markov chain of the randomly selected nodes in the graph. Each time, a neighbor of the current node is selected randomly. After numbers of iterations, the result will converge.

CHAPTER 4
SECOND-ORDER COSIMRANK IN ONLINE SOCIAL NETWORKS

Measuring the similarity between nodes is challenging in social networks. The SimRank and CoSimRank are techniques widely used to calculate the similarity of two nodes in a social graph. They can be applied to many applications such as recommending friends and detecting communities in social networks. Both SimRank and CoSimRank are based on random walk and only consider first-order transition probabilities, in which the next node to visit in random walk only depends on the current node, like a Markov chain. However, in many real-world situations, simply considering the current node may not be enough. Previously visited node may provide extra information for measuring similarities. In this chapter, we propose a novel similarity measure technique by investigating CoSimRank to take advantage of the second-order information in a random walk process [20].

4.1 Introduction

Social networks have received incredible growth in recent years. Facebook, Twitter and Instagram are among the most popular social networking platforms. Social graphs or networks are often used to depict personal relations of Internet users in these platforms. In a social network, the nodes represent individuals in the network, and the edges represent the social interactions between them [21, 22]. It is crucial to mine useful knowledge from the huge and complex social network graphs efficiently. For example, measuring the similarity is a fundamental step for many applications like querying and ranking, community detection [23, 24, 25] and link prediction [18, 26]. To measure the similarity of nodes, a common

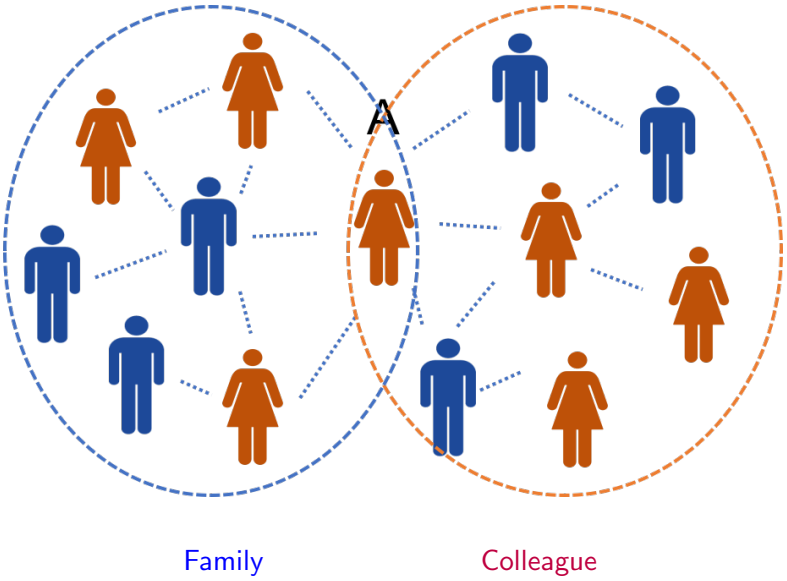

Figure 4.1: A subgraph of relationship

procedure is to measure the proximity between nodes. Random walk has been proven to be a simple yet effective method for proximity measure. The main idea of random walk is to allow a surfer to randomly traverse a graph based on the transition probability of the graph. Some of the popular random walk based proximity measure approaches include PageRank [12], random walk with restart [13] and SimRank [14].

However, the existing random walk based proximity measure approaches were developed using the first-order Markov model, meaning that they do not consider any previous nodes other than the current node. Figure 4.1 shows user connections in a social network. Each node represents a user. If two nodes are connected by an edge, it means these two users are friends. Since the community is a group of nodes that are similar to each other while dissimilar with nodes in other groups, there are two communities in the figure. The left part of the figure is for families, while the right part of the figure is for colleagues. Most of the

current existing random walk based approaches do not consider where the surfer came from. Suppose that the random surfer came from A's family community. Based on the first-order proximity measure, the probability of visiting the nodes in colleague community is the same as the nodes in the family community because the surfer has no memory on the previous nodes. However, in real-world situations, people exploring family connections could have a higher probability of visiting another family member.

SimRank is one of the popular random walk based proximity measures. It was developed based on the intuitive assumption that two nodes are similar if they are referenced by similar nodes [14]. SimRank can be applied to any domain as long as there are enough relevant relationships between objects. However, one of the major issues of SimRank is that it is computationally expensive, as it requires calculation of all other SimRank values before determining any value between two nodes. Some work tried to improve its efficiency [27, 28, 29]. S. Rothe and H. Schütze designed CoSimRank [30], which is much faster than SimRank because it does not need to compute the similarity values of the whole graph in order to calculate the value of two nodes. W. Yu and J. McCann designed Co-Simmate to improve its speed, which reduces the time of computing all pairs of CoSimRank to just $\mathcal{O}(\log_2(\log(1/\epsilon))n^3)$ while still retaining the accuracy of the model [31]. Y. Wu et al. developed the second-order random walk in PageRank, random walk with restart, SimRank and SimRank* [32, 33]. In [34], the authors applied CoSimRank on dynamic graph. However, to our best knowledge, there is no work developing the second-order structure of CoSimRank.

In this paper, we propose the second-order CoSimRank, which will take advantages of

the second-order structure such as trigrams to facilitate exploring the community structures and social networks. Standard adjacency matrices can only represent the first-order graph information and define node-to-node transition probabilities. If we consider the already visited node, we will need to build a tensor [35]. Using tensors will increase the computational difficulty. Thus, we use edge-to-edge transition probabilities. Since the incidence matrices of the graph can represent such probabilities [36], we propose matrix representations for the second-order measures for CoSimRank.

The rest of the chapter is organized as follows. Section II introduces the related work. Section III presents the problem formulation. In Section IV, the second-order CoSimRank method and related analysis are elaborated. Section V compares our method and with the baseline methods such as first-order CoSimRank. Finally, conclusions are drawn in Section VI.

4.2 CoSimRank and Second-order Random Walk

In this section, we introduce the background knowledge of CoSimRank. We first introduce the first-order CoSimRank then the second-order random walk. Table 4.1 shows the main symbols we will use in this paper.

4.2.1 CoSimRank

CoSimRank starts with a probability distribution at each time point in the first-order random walk. Suppose the random surfer starts from node i and randomly explores the network following the first-order random walk. Let $\mathbf{r}^{t,i}$ represent the probability distribution vector

at time point t when the random surfer starts from node i. Since the random surfer starts from node i at time point t ($t = 0$), we have that $\mathbf{r}^{0,i} = \mathbf{q}^i$, where

$$[\mathbf{q}^i]_k = \begin{cases} 1 & \text{if } k = i, \\ 0 & \text{if } k \neq i. \end{cases}$$

For time point t ($t \geq 1$), let $\mathbf{P}$ represent the transition matrix, we have

$$\mathbf{r}^{t,i} = \mathbf{P}^{\mathrm{T}}\mathbf{r}^{t-1,i} \tag{4.1}$$

Thus, $\mathbf{r}^{t,\,i}$ can be represented as

$$\mathbf{r}^{t,i} = \begin{cases} \mathbf{q}^i & \text{if } t = 0, \\ \mathbf{P}^{\mathrm{T}}\mathbf{r}^{t-1,i} & \text{if } t \geq 1. \end{cases}$$

CoSimRank is defined by using the probability distribution vectors $\mathbf{r}_{t,i}\mathbf{r}^{t,\,i}$. Specifically, the CoSimRank values $\mathbf{s}_{(i,j)}$, between node i and j is defined as

$$\mathbf{s}_{(i,j)} = \sum_{t=0}^{\infty} c^t \cdot \left\langle \mathbf{r}^{t,i}, \mathbf{r}^{t,j} \right\rangle \tag{4.2}$$

where the operator $\langle \cdot, \cdot \rangle$ represents the inner product

$$\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{k=1}^{n} x_k \cdot y_k \tag{4.3}$$

and $c (0 < c < 1)$ is a decay factor.

CoSimRank is similar to SimRank in terms of definition [30]. The experimental results demonstrated that CoSimRank is more accurate than both SimRank and personalized

(a) original graph (b) the line graph

Figure 4.2: An example graph and its line graph

PageRank [30].

4.2.2 Second-order Random Walk

In first-order random walk, a node-to-node transition probability is used to determine where the surfer will go. In the second-order random walk, edge-to-edge transition probability is used [32]. It can be treated as the node-to-node transition in the line graph of the original graph.

Let $p_{i,j}$ be the probability of the surfer visiting node j from node i and r_j^t be the probability of the surfer visiting node j at time t. We have $r_j^t = \sum_{i \in I_j} p_{i,j} \cdot r_i^{t-1}$ where I_j is the set of in-neighbors of j. When we deal with the second-order random walk, let $p_{i,j,k}$ be the probability that the surfer is currently at node j and on the way to node k, after previously visiting node i. Further assume that the probability of visiting j from i is u, and the probability of visiting k from j is v. The node to node probability $p_{i,j,k}$ can then be represented by edge-to-edge probability $p_{u,v}$. In Figure 4.2a, the probability of $p_{1,4,2}$ is the same as the probability $p_{b,e}$ in Figure 4.2b.

Table 4.1: Main symbols for second-order CoSimRank

symbols	definitions
$G(V, E)$	directed graph G with node set V and edge set E
I_i , O_i	set of in-/out-neighbor nodes of node i
n, m, σ	number of nodes; number of edges; $\sigma = \sum_{i \in V} \lvert I_i \rvert \cdot \lvert O_i \rvert$
$\mathbf{B}$	$n \times m$ incidence matrix, $[\mathbf{B}]_{i,u} = 1 : u$ is an out-edge of i
$\mathbf{E}$	$m \times n$ incidence matrix, $[\mathbf{E}]_{u,i} = 1 : u$ is an in-edge of i
$w_{i,j}$, w_i	weight of edge (i, j); out-degree of i: $w_i = \sum_{j \in O_i} w_{i,j}$
$\mathbf{W}$	$m \times m$ diagonal matrix, $[\mathbf{W}]_{u,u} = w_{i,j}$ if edge $u = (i, j)$
$\mathbf{D}$	$n \times n$ diagonal matrix, $[\mathbf{D}]_{i,i} = w_i$
$p_{i,j}$	transition probability from node i to j
$p_{i,j,k}$	transition prob. from j to k if the surfer came from i
$p_{u,v}$	transition prob. from edge u to v, $p_{(i,j),(j,k)} = p_{i,j,k}$
$\mathbf{P}$	$n \times n$ node-to-node transition matrix, $[\mathbf{P}]_{i,j} = p_{i,j}$
$\mathbf{H}$	$n \times m$ node-to-edge transition matrix, $[\mathbf{H}]_{i,(i,j)} = p_{i,j}$
$\mathbf{M}$	$m \times m$ edge-to-edge transition matrix, $[\mathbf{M}]_{u,v} = p_{u,v}$
$r_{i,j}$, r_i	$r_{i,j}$: proximity value of node i w.r.t. node j; $r_i = r_{i,q}$
$\mathbf{r}, \mathbf{R}$	$\mathbf{r}$: $n \times 1$ vector, $\mathbf{r}_i = r_i$; $\mathbf{R}$: $n \times n$ matrix, $[\mathbf{R}]_{i,j} = r_{i,j}$
s_u, $s_{(i,j)}$	proximity value of edge u or (i, j) w.r.t. query node q
$s_{u,v}$	proximity value between edges u and v
$\mathbf{s}, \mathbf{S}$	$\mathbf{s}$: $m \times 1$ vector, $\mathbf{s}_u = s_u$; $\mathbf{S}$: $m \times m$ matrix, $[\mathbf{S}]_{u,v} = s_{u,v}$

In the second-order random walk, the incidence matrix is used instead [32]. Incidence matrix is the node-to-edge matrix representing the relationship of nodes and edges. Let $\mathbf{B}$

and $\mathbf{E}$ be the out-edges and in-edges incidence matrices respectively. If element $[\mathbf{B}]_{i,u}$ is 1, the edge u is an out-edge of node i. If element $[\mathbf{E}]_{u,i}$ is 1, the edge u is an in-edge of node i. Otherwise the elements are 0. For example in Figure 4.2a, the incidence matrices are

$$\mathbf{B} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{ccccc} \mathsf{a} & \mathsf{b} & \mathsf{c} & \mathsf{d} & \mathsf{e} \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}, \quad \text{and} \quad \mathbf{E}^{\mathsf{T}} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{ccccc} \mathsf{a} & \mathsf{b} & \mathsf{c} & \mathsf{d} & \mathsf{e} \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix} \end{array}.$$

Now let $\mathbf{H}$ be the node-to-edge transition probability matrix. $\mathbf{H}$ can also be represented by the equation $\mathbf{H} = \mathbf{D}^{-1}\mathbf{B}\mathbf{W}$, where $\mathbf{W}$ is the diagonal matrix with $[\mathbf{W}]_{u,u}$ denoting the weight of edge u. $\mathbf{P} = \mathbf{H}\mathbf{E}$ can then represent the node-to-node transition probability matrix.

4.3 Second-order CoSimRank

In this section, we present our second-order CoSimRank. The matrix form and convergence properties will be given, which is followed by a comparison of SimRank and CoSimRank.

4.3.1 Second-order CoSimRank

Suppose the random surfer starts from node i and randomly explore the network following the second-order random walk. Let $\mathbf{r}^{t,i}$ represent the probability distribution vector at time point t when the random surfer starts from node i. Since the random surfer starts from node i at time point $t = 0$, we have that

$$\mathbf{r}^{0,i} = \mathbf{q}^{i} \tag{4.4}$$

where

$$[\mathbf{q}^i]_k = \begin{cases} 1 & \text{if } k = i, \\ 0 & \text{if } k \neq i. \end{cases}$$

For the time point $t = 1$, we have that

$$\mathbf{r}^{t,i} = \mathbf{P}^\mathsf{T}\mathbf{q}^i \tag{4.5}$$

For the time point $t \geq 2$, we have that

$$\mathbf{r}^{t,i} = \mathbf{E}^\mathsf{T}(\mathbf{M}^\mathsf{T})^{t-1}\mathbf{H}^\mathsf{T}\mathbf{q}^i \tag{4.6}$$

where $\mathbf{E}$ is in-edge incidence matrix. $\mathbf{M}$ is the edge-to-edge transition matrix. $\mathbf{H}$ is the node-to-edge transition matrix.

Thus, $\mathbf{r}^{t,i}$ for second-order CoSimRank can be represented by

$$\mathbf{r}^{t,i} = \begin{cases} \mathbf{q}^i & \text{if } t = 0, \\ \mathbf{P}^\mathsf{T}\mathbf{q}^i & \text{if } t = 1, \\ \mathbf{E}^\mathsf{T}(\mathbf{M}^\mathsf{T})^{t-1}\mathbf{H}^\mathsf{T}\mathbf{q}^i & \text{if } t \geq 2. \end{cases}$$

CoSimRank can be defined using the probability distribution vectors $\mathbf{r}^{t,i}$. Specifically, the second-order CoSimRank values $\mathbf{s}_{(i,j)}$ between node i and j are defined as

$$\mathbf{s}_{(i,j)} = \sum_{t=0}^{\infty} c^t \cdot \left\langle \mathbf{r}^{t,i}, \mathbf{r}^{t,j} \right\rangle \tag{4.7}$$

where the operator $\langle \cdot, \cdot \rangle$ represents the inner product

$$\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{k=1}^{n} x_k \cdot y_k \tag{4.8}$$

Let $\mathbf{R}^{(t)}$ represent the corresponding $\mathbf{R}$ at time point t. We know that $\mathbf{R}^{(0)} = \mathbf{I}$ where $\mathbf{I}$ is the identity matrix, and the series form can be written as

$$\mathbf{R} = \sum_{t=1}^{\infty} c^t \mathbf{H}\mathbf{M}^{t-1}\mathbf{E}\mathbf{E}^{\mathsf{T}}(\mathbf{M}^{\mathsf{T}})^{t-1}\mathbf{H}^{\mathsf{T}} + \mathbf{I} \tag{4.9}$$

We find:

$$\begin{aligned} \mathbf{R}^{(1)} &= c\mathbf{H}\mathbf{E}\mathbf{E}^{\mathsf{T}}\mathbf{H}^{\mathsf{T}} + \mathbf{R}^{(0)} \\ \mathbf{R}^{(2)} &= c^2\mathbf{H}\mathbf{M}\mathbf{E}\mathbf{E}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}\mathbf{H}^{\mathsf{T}} + \mathbf{R}^{(1)} \\ \mathbf{R}^{(3)} &= c^3\mathbf{H}\mathbf{M}^2\mathbf{E}\mathbf{E}^{\mathsf{T}}(\mathbf{M}^{\mathsf{T}})^2\mathbf{H}^{\mathsf{T}} + \mathbf{R}^{(2)} \\ &\cdots \end{aligned} \tag{4.10}$$

Generally, for $t \geq 1$ we have

$$\mathbf{R}^{(t)} = c^t\mathbf{H}\mathbf{M}^{(\mathrm{t\text{-}1})}\mathbf{E}\mathbf{E}^{\mathsf{T}}(\mathbf{M}^{\mathsf{T}})^{(\mathrm{t\text{-}1})}\mathbf{H}^{\mathsf{T}} + \mathbf{R}^{(t-1)} \tag{4.11}$$

4.3.2 Convergence Properties

We now show that the values of second-order CoSimRank exist and are unique.

Theorem 1. *There exists a unique solution to the second-order CoSimRank.*

Proof. We can see in Eq. (4.2) that the value of single pair CoSimRank is monotonically increasing as follows.

$$\sum_{t=0}^{\infty} c^t \cdot \left\langle \mathbf{r}^{t,i}, \mathbf{r}^{t,j} \right\rangle \leq \sum_{t=0}^{\infty} c^t$$

In specific, if $\langle \mathbf{r}^{t,i}, \mathbf{r}^{t,j} \rangle$ is 1 in every iteration, $\mathbf{s}_{(i,j)} = \sum_{t=0}^{\infty} c^t$. If $\langle \mathbf{r}^{t,i}, \mathbf{r}^{t,j} \rangle \leq 1$ in every iteration, $\mathbf{s}_{(i,j)} \leq \sum_{t=0}^{\infty} c^t$. By Cauchy-Schwarz inequality, the upper bound of the inner product of two vectors is

$$\langle u, v \rangle \leq ||u||\, ||v||$$

In our case, $\mathbf{r}^{t,i}$ and $\mathbf{r}^{t,j}$ are two probability vectors, so $||\mathbf{r}^{t,i}||\, ||\mathbf{r}^{t,j}|| = 1$ and $\langle \mathbf{r}^{t,i}, \mathbf{r}^{t,j} \rangle \leq 1$. Therefore,

$$\mathbf{s}_{(i,j)} \leq \sum_{t=0}^{\infty} c^t$$

As $\sum_{t=0}^{\infty} c^t = \frac{1}{1-c}$, we can express the above inequality as

$$\mathbf{s}_{(i,j)} \leq \frac{1}{1-c}$$

Together with the fact that $\mathbf{s}_{(i,j)}$ is monotonically increasing, it will finally converge.

As shown by the matrices in Eq. (4.9), a unique graph generates unique matrices $\mathbf{H}$, $\mathbf{M}$, $\mathbf{E}$. Given the uniqueness of the matrices, the value $\mathbf{R}$ is also unique. □

4.3.3 Comparison of Second-order SimRank and Second-order CoSimRank

The first-order SimRank can be written as

$$\mathbf{R} = (c\mathbf{P}\mathbf{R}\mathbf{P}^{\mathsf{T}}) \vee \mathbf{I}$$

where $\mathbf{I}$ is the identity matrix, $\vee$ denotes the element-wise maximum, i.e., (i, j) entry of the matrix $\mathbf{A} \vee \mathbf{B}$ is given by $max\{\mathbf{A}_{i,j}, \mathbf{B}_{i,j}\}$, $\mathbf{P}$ is the transition matrix.

As the element-wise maximum is a non-linear operation, $\mathbf{R}$ can be rewritten as

$$\mathbf{R} = c\mathbf{P}\mathbf{R}\mathbf{P}^{\mathsf{T}} + \mathbf{D}$$

where $\mathbf{D}$ the diagonal correction matrix with $(1-c) \leq \mathbf{D}_{i,i} \leq 1$. The computation of $\mathbf{D}$ is also costly. So we usually use a simple approximation $\mathbf{D} \approx (1-c)\mathbf{I}$ [32] and we have

$$\mathbf{R} = c\mathbf{P}\mathbf{R}\mathbf{P}^{\mathsf{T}} + (1-c)\mathbf{I}$$

The series form can be represented by

$$\mathbf{R} = (1-c)\sum_{t=0}^{\infty} c^t \mathbf{P}^t (\mathbf{P}^{\mathsf{T}})^t \tag{4.12}$$

The series form of first-order CoSimRank can be written as

$$\mathbf{R} = \sum_{t=0}^{\infty} c^t \mathbf{P}^t (\mathbf{P}^{\mathsf{T}})^t \tag{4.13}$$

From the equations for SimRank and CoSimRank, we can find they differently initialize the similarities along the diagonal. SimRank sets each diagonal entries to one at each iteration while CoSimRank adds one.

In second-order, the recursive equation of second-order SimRank is as the following:

$$\begin{cases} \mathbf{S} = c\mathbf{M}\mathbf{S}\mathbf{M}^{\mathsf{T}} + (1-c)\mathbf{E}\mathbf{E}^{\mathsf{T}} \\ \mathbf{R} = c\mathbf{H}\mathbf{S}\mathbf{H}^{\mathsf{T}} + (1-c)\mathbf{I} \end{cases}$$

In series form, second-order SimRank can be represented by

$$\mathbf{R} = (1-c)\sum_{t=1}^{\infty} c^t \mathbf{H}\mathbf{M}^{t-1}\mathbf{E}\mathbf{E}^{\mathsf{T}}(\mathbf{M}^{\mathsf{T}})^{t-1}\mathbf{H}^{\mathsf{T}} + (1-c)\mathbf{I} \tag{4.14}$$

We have the element-wise form of second-order CoSimRank in Eq. (4.9). By comparing Eq. in (4.9) and (4.14), we can observe that the two equations are similar since they are both in first-order.

4.4 Second-order CoSimRank Calculating Algorithm

In this section, we introduce the algorithm to calculate a single-source second-order CoSimRank score. Single-source second-order CoSimRank is querying from a node v_i requesting for the second-order CoSimRank score between v_i and every other node.

Given a recursive equation and the condition that the returned matrix or vector will converge, the power iteration can be used to find the final converged matrix or vector by running the equation over and over. Below is the stopping criterion for vector

$$||\mathbf{r}^t - \mathbf{r}^{t-1}||_1 < \epsilon$$

and for matrix

$$||\mathbf{A}^t - \mathbf{A}^{t-1}||_1 < \epsilon$$

where $||\mathbf{r}||_1 = \sum_i |r_i|$ and $||\mathbf{A}||_1 = \sum_{i,j} |A_{i,j}|$ denote the sum of absolute values, ϵ is a small positive value.

Power iteration is easy to implement when we know that the matrix or vector would finally converge. In our experiment, we use Eq. (4.11) to perform power iteration and use the value of $R^{(t)}$ to compute $R^{(t+1)}$. When the above criterion is met, we stop the iterations. However, the power iteration can take extended time before it terminates. Therefore, we

develop another method to perform the computing experiments as shown in Algorithm 1.

Algorithm 1 Single-source algorithm for the second-order CoSimRank

Input : $G(V, E)$, query node $\mathbf{q}$, decay factor c, number of iterations β, transition matrices $\mathbf{M}$ and $\mathbf{H}$, incidence matrix $\mathbf{E}$
Output : proximity vector $\mathbf{r}$

1: $\mathbf{r} \leftarrow \mathbf{q}$
2: $\mathbf{r}' \leftarrow \mathbf{H}^{\mathrm{T}}\mathbf{q}$
3: **for** $t \leftarrow 1$ to β **do**
    $\mathbf{r} \leftarrow \mathbf{r} + c^t\mathbf{H}\mathbf{M}^{t-1}\mathbf{E}\mathbf{E}^{\mathrm{T}}\mathbf{r}'$
    $\mathbf{r}' \leftarrow \mathbf{M}^{\mathrm{T}}\ \mathbf{r}'$
return r

4.4.1 Single-source Algorithm

The single-source values of second-order CoSimRank can be computed by the following equation

$$\mathbf{r} = \sum_{t=1}^{\beta} c^t\mathbf{H}\mathbf{M}^{t-1}\mathbf{E}\mathbf{E}^{\mathrm{T}}(\mathbf{M}^{\mathrm{T}})^{t-1}\mathbf{H}^{\mathrm{T}}\mathbf{q} + \mathbf{q}$$

where β is the number of iteration, $\mathbf{q}$ is the query node. $\mathbf{r}$ is a vector containing second-order CoSimRank values of query node $\mathbf{q}$ with any other nodes in the graph. As shown in Algorithm 1, it starts with initializing $\mathbf{r}$ with query node $\mathbf{q}$. The right hand side of the equation is calculated by maintaining $\mathbf{r}' = (\mathbf{M}^{\mathrm{T}})^{t-1}\mathbf{H}^{\mathrm{T}}\mathbf{q}$.

Time complexity: Time complexity of the single–source algorithm is $O(\beta\rho n)$. To calculate the values of all node in one iteration, it takes $O(\rho n)$. To iterate β times, it takes $O(\beta\rho n)$.

Space complexity: The space complexity for single–source algorithm is $O(\beta n^2)$. As mentioned earlier, single-pair value takes $O(\beta n)$ space. When we need to calculate all n nodes, it takes $O(\beta n^2)$.

4.4.2 Top-k nodes Algorithm

The top-k nodes algorithm is based on the single-source algorithm. After Algorithm 1 being called, we get the CoSimRank value $\mathbf{r}$. Then, we sort the nodes and retrieve the top-k nodes in the list.

Time complexity: As we know, the time complexity of single-source algorithm is $O(\beta\rho n)$. After getting $\mathbf{r}$ from the algorithm, we sort the nodes. Sorting the nodes takes $O(n \log k)$. Overall, it takes $O(\beta\rho n + n \log k)$. However, $O(\beta\rho n)$ is much greater than $O(n \log k)$. Hence, to get top-k nodes, it takes $O(\beta\rho n)$.

Space complexity: As mentioned earlier, the space complexity of getting single-source is $O(\beta n^2)$. If we only get the top-k nodes, it will take $O(\beta nk)$ space.

4.5 Performance Evaluation

In this section, we present our experiments with the ground truth datasets and synthetic datasets. We use a normalized distributed cumulative cost gain (NDCG) statistical model to measure the accuracy of the CoSimRank algorithms. We use Facebook and Twitter API to get the real-world datasets. The datasets used for comparing the computing complexity of CoSimRank are created with a graph generator, GTGraph [37].

4.5.1 Datasets

In this experiment, we try to compare the accuracy and execution time of first-order and second-order CoSimRank with real-world datasets. For the Facebook data, we programmed a python project to access a random user and create an edge list of friends for the experimental

dataset. Similarly, we access a random Twitter user and create an edge list of followers for the Twitter experimental dataset.

One should note that in Facebook, the network is an undirected network because if user A is a friend of user B, it must be the same in the other way around. After we crawl the data on Facebook, we transform the directed Facebook dataset into undirected by copying the dataset and switching the two nodes in every edge. For example, if there is an edge (i, j) appearing in the original dataset, there should be another edge (j, i) appearing in the newly created dataset. In Twitter, the network is directed. If user A is a follower of user B, it is not necessary that user B is a follower of A.

4.5.2 Link Prediction

Facebook users and Twitter users are queried as the datasets for this experiment. A group of 46,008 users and 385,641 edges for Facebook, 12,150 users and 39,447 edges for Twitter are used. The data crawled before March 2018 is used for training, and the data crawled in September 2018 is used for testing. The autoregressive model is chosen to calculate the edge-to-edge transition matrix, where α is set at 0.5. To evaluate the accuracy, we define the precision as:

$$precision = \frac{k'}{k}$$

where k' is the number of correctly predicted edges that exactly appeared in the original dataset, k is the number of edges used as testing data. We randomly pick 10^3 query edges and generate the average. The training data is used to calculate the ranking values and the

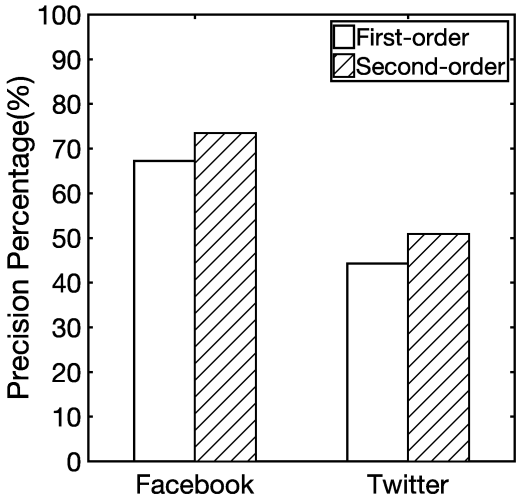

Figure 4.3: Link prediction precision for Facebook and Twitter data

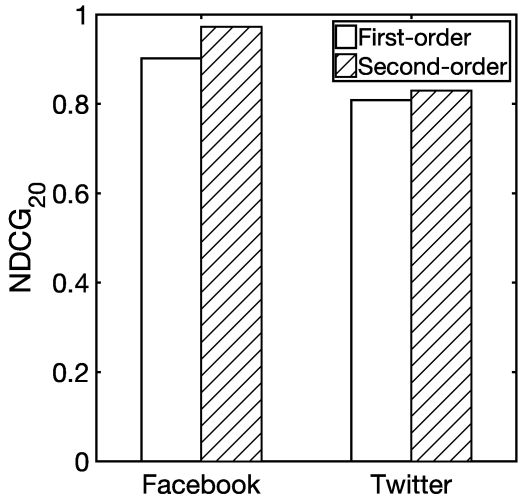

Figure 4.4: Facebook and Twitter top-k query accuracy

precision is shown in Figure 4.3.

In Figure 4.3, we can see that no matter in the Facebook or Twitter dataset, the second-order CoSimRank outperforms the first-order. The second-order CoSimRank improves the precision by approximately 7% for Facebook data and 19% for Twitter data. This is due to the fact that the second-order CoSimRank considers the previously visited node, which can take advantages of the correlation among the visited nodes that are within two hops. The precision for Facebook data is higher than Twitter data, which shows a larger size of data can improve the precision value.

4.5.3 Top-k Query

In this experiment, a random user of Facebook and Twitter is selected, and the top twenty relevant friends are chosen. The technique used to rank importance is on a scale of one to five (five being the most relevant). We then observe how many friends that person has and

whether that friend is another person or non-person (i.e., movie, company, product, etc.). A non-person decreases relevance ranking. From these observed friends, another list of twenty is created out of randomly selected friends. If a selected friend had no further friends, he is treated as a dangling node and his relevance is decreased.

The computation of NDCG of the first and second-order CoSimRank is also compared. The discounted cumulative gain (DCG) measures the quality of the results in a ranked list, where items in that list are ranked by the scale of one to five. It checks the relevance for each randomly selected user or node.

Normalized DCG (NDCG) is defined to calculate this measure across the independent queries, the result lists for which might all be of different length. NDCG is computed as:

$$\mathrm{NDCG}_p = \frac{DCG_p}{IDCG_p}$$

If more than one user is selected, the NDCG values are averaged to obtain an accuracy value, whereas p is the number of chosen users or nodes. The results shown in Figure 4.4, indicate the CoSimRank second-order is more accurate in ranking the top twenty friends of a friend on Facebook. It also indicates an improvement in ranking the top twenty Twitter followers. This is because the second-order measures can better capture the information among users and significantly improve the accuracy of the results.

4.5.4 Computational Runtime

The measure of computational complexity is taken using synthetic data in this experiment. The synthetic dataset is generated using GTGraph. Table 4.2 shows the generated dataset

Table 4.2: Synthetic data characteristics

Nodes $\times 2^{16}$	**Edges** $\times 10^5$
1	1
2	2
4	4
8	8

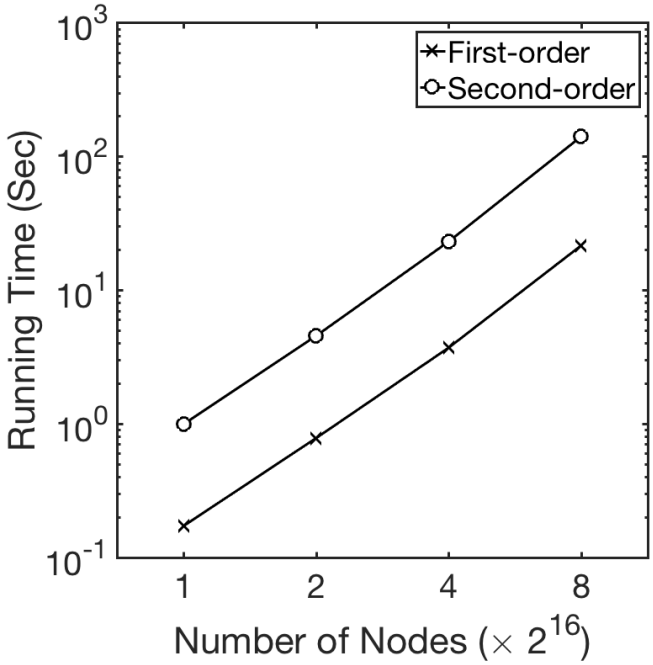

Figure 4.5: Computational time complexities

and its statistics. The runtime is collected for each process of CoSimRank with first and second-orders. Figure 4.5 shows the computational running time with respect to the number of nodes. We can find that for the second-order CoSimRank, it is slightly slower than the first-order CoSimRank. This is because we use the edge-to-edge information to calculate the value, and in a real-world application, the number of edges is usually larger than the number of nodes.

4.6 Chapter Summary

Designing effective proximity measures for large graphs in the social network is a critical and computationally challenging task. With the recent introduction of CoSimRank, we can efficiently calculate the similarity values of two nodes. In this paper, we have proposed a second-order CoSimRank measure scheme which takes the previously visited node into consideration. We have presented the rigorous theoretical foundations for the second-order CoSimRank and developed second-order algorithms for second-order CoSimRank. The experimental results have demonstrated that the proposed second-order CoSimRank measures outperform the first-order CoSimRank in various applications. The simulation results of the random graph have also shown that the time complexity of the second-order CoSimRank is slightly higher than that of the first-order CoSimRank.

CHAPTER 5
INFLUENCERS ANALYSIS IN ONLINE SOCIAL NETWORKS

Nowadays, people are exposed to tons of fake information or misleading posts in Online Social Networks. Celebrities sometimes intentionally create misleading posts in OSNs to guide people for commercial or marketing purposes. The intentional phrases from such posts can affect the online rating and even lead to a frenzy shopping. That's part of the reasons that the top social media influencers are targeted by merchants to help promote products. The Federal Trade Commission (FTC) requires that all sponsored posts must be clearly disclosed. However, many influencers do not do that. As a result, people may be misled by the undisclosed sponsorship. In this chapter, we explore the credibility of posts on Instagram and analyze if an influencer complies with the FTC requirements.

5.1 Introduction

Online Social networks (OSNs) have become prevalent in people's daily life. They allow customers to interact with others, share their consumption experiences and make product recommendations. The ability of digital platforms to help build and utilize the interactions between users has become a critical factor for the successful communication networks. For example, Instagram, as a popular digital media sharing platform [38], is commonly used to increase social self-worth for users and promote merchants for businesses. It not only provides a good platform for customers to post photos, videos and texts but also gives merchants a channel to apply the socially engaged marketing strategies.

To make the advertisement less intrusive, merchants transfer their promotion to OSNs

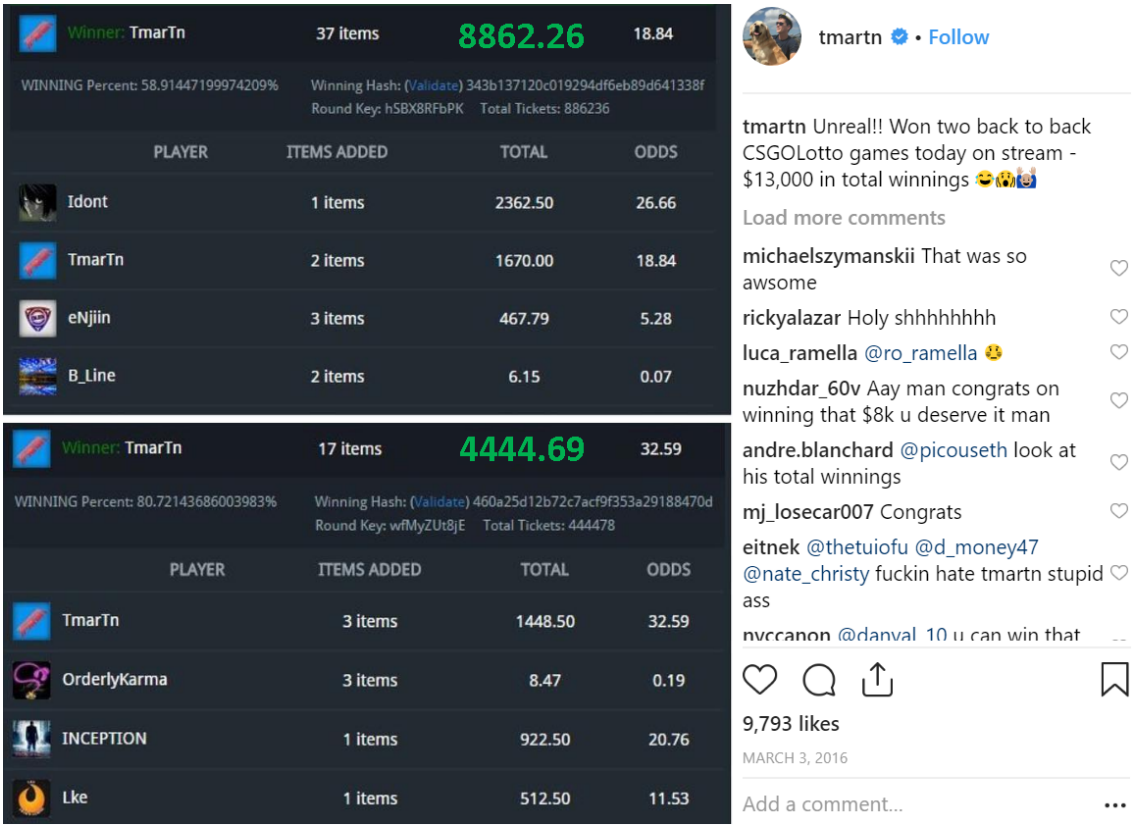

Figure 5.1: An example of undisclosed sponsored post on Instagram

currently. Typically, there are two main types of marketing strategies: automating actions and non-automating actions. Automating actions mean that merchants use bots to artificially increase the popularity and engagement of their products. For example, by purchasing likes and complimentary comments, it may give customers the impression that the product is popular and useful. However, OSN platforms constantly monitor bots in order to make their platform more trustful and reflect real people's activities. Brands or advertisers may have the risk of their accounts or posts (boosted by bots) being warned or suspended. Therefore, lots of merchants choose non-automating actions to escape from the monitoring. Non-automating actions mean that merchants may pay highly connected and trusted "in-

fluencers" to post contents and advertise their products implicitly. In this work, we define a social influencer as an Instagram user with more than 10,000 followers. If a post does not disclose that the user was paid by a sponsor, the user is considered as a black hat influencer [39]. The Federal Trade Commission (FTC) has declared that all sponsored posts must be clearly disclosed. Recently, they strengthened the enforcement actions against users who are not disclosing sponsored posts [40]. OSNs, such as Instagram, have added tools that allow users to explicitly disclose sponsorship so that they can comply with the FTC rules. However, many highly connected or trusted influencers are still not disclosing their sponsored posts. For instance, Figure 5.1 shows an undisclosed sponsored post on Instagram. The right side shows the user "tmartn", post text, comments, and number of likes. The left side show the post images. The numbers "8862.26" and "4444.69" in the post image represent the money the user won and they are attractive. The post text not only emphasizes the money the user "tmartn" won, but also expresses the excitement for this game. This post and its publisher are identified by FTC's website as deceptively endorsing the online gambling site CSGOLotto without disclosing that the user "tmartn" owns the company [40, 41]. This situation is common because the influencers want to make profit by using their large number of followers.

Previous work on analyzing the sponsored posts focuses on two main directions: sponsored search and auction analysis [42, 43], and sponsored content and service market analysis [44, 45, 46]. Sponsored search refers to the search result mixed with advertisements [42, 47]. Sponsored content and service market analysis mainly investigates the pricing strategies for

the marketing campaigns. None of them are considering sponsorship disclosure detection or analyzing the integrity of social influencers in OSNs. In this work, we propose an effective Undisclosed Sponsored Post Detection (USPD) framework based on the ensemble machine learning classifier. This framework takes advantages of the texts and images in Instagram posts to enhance the detection accuracy. The contributions are summarized as follows:

- To our best knowledge, our work is the first to analyze the credibility of Instagram posts from the sponsored content perspective.
- The proposed framework not only extracts the traditional linguistic features but also makes use of the effective community related features and image based features.
- For model training, we exploit the auto-labeled ground-truth data to significantly improve our model performance.

The rest of the paper is organized as follows. Section 5.2 introduces the sponsored post characterizations in OSNs. Section 5.3 presents the problem formulation and the proposed framework. Section 5.4 elaborates the experimental setting, results and analysis. Section 6.7 draws the conclusion.

5.2 Sponsored Posts in OSNs

With more strategy shifting from traditional newspaper or television to OSNs, brands and merchants pay a lot of attention to influencers' blog posts. Personal blogs have been actively used to promote products. Even though some blog posts are written to express personal emotions without any marketing purposes, many of posts are sponsored by third parties.

Sponsored posts refer to consumer-generated blog posts that are sponsored by brands [48]. There are two types of sponsored posts on Instagram: paid Instagram ads and user posts with disclosure. Paid Instagram ads mean that brands or merchants pay Instagram to display their posts. These posts are marked as "sponsored" and are displayed in users' feeds. User posts with disclosure mean that posters explicitly disclose the partnership or connection with the brand in their post texts. They can place a hashtag such as #sponsored or #ad in the text of the sponsored Instagram posts [49]. According to FTC, these are the only two hashtags that make the partnership between poster and brand clear [49].

Some brands or companies secretly pay users with a lot of followers to promote their products to a larger audience. The connection between the publisher and the company is underground. In these cases, sponsorship is not sufficiently disclosed to OSN users. As a result, there are a large number of undisclosed sponsored posts on Instagram, which could largely affect the trust between people and products. In this paper, we define two types of sponsored posts as follows:

Disclosed Sponsored Post (DSP): A disclosed sponsored post is a post that is sponsored by commercial companies and contains the #sponsored or #ad hashtag in its text.

Undisclosed Sponsored Post (USP): An undisclosed sponsored post is a post that is sponsored by commercial companies but contains neither the #sponsored nor the #ad hashtag in its text.

5.3 Undisclosed Sponsored Post Detection

In this section, we formally define the sponsored post detection problem and propose our Undisclosed Sponsored Post Detection (USPD) framework.

5.3.1 Problem Definition

Let s denote an Instagram Post. It consists of two major components: *Publisher* and *Content*. The publisher $\vec{p}_s$ consists of a set of profile features, such as name, user id and description. The content $\vec{c}_s$ includes a set of attributes from an Instagram post such as caption, image, location, etc. We define a *Social Engagement* set $\mathbf{S}_p = \left\{s_p^1, s_p^2, ..., s_p^n\right\}$ to represent the historical posts of a publisher p, where s_p^n is the n-th post from p. Further, let $\mathbf{S} = \bigcup_p \mathbf{S}_p$ be the set of all posts. Table 5.1 shows the main symbols we will use in this paper.

Undisclosed Sponsored Post Detection: Given an Instagram post $s \in \mathbf{S}$, determine whether s is an undisclosed sponsored post (USP) or not.

Undisclosed Sponsored Post Detection is a binary classification problem. It is challenging as an USP does not contain #sponsored or #ad hashtags (as a DSP does). However, since both USP and DSP are commercial ads, they bear some substantial similarities. This motivates us to use the available DSPs to help detect the USPs. Specifically, we can use the scraped DSPs to train machine learning models for USP detection, as the proposed Undisclosed Sponsored Post Detection (USPD) framework in Fig. 5.2.

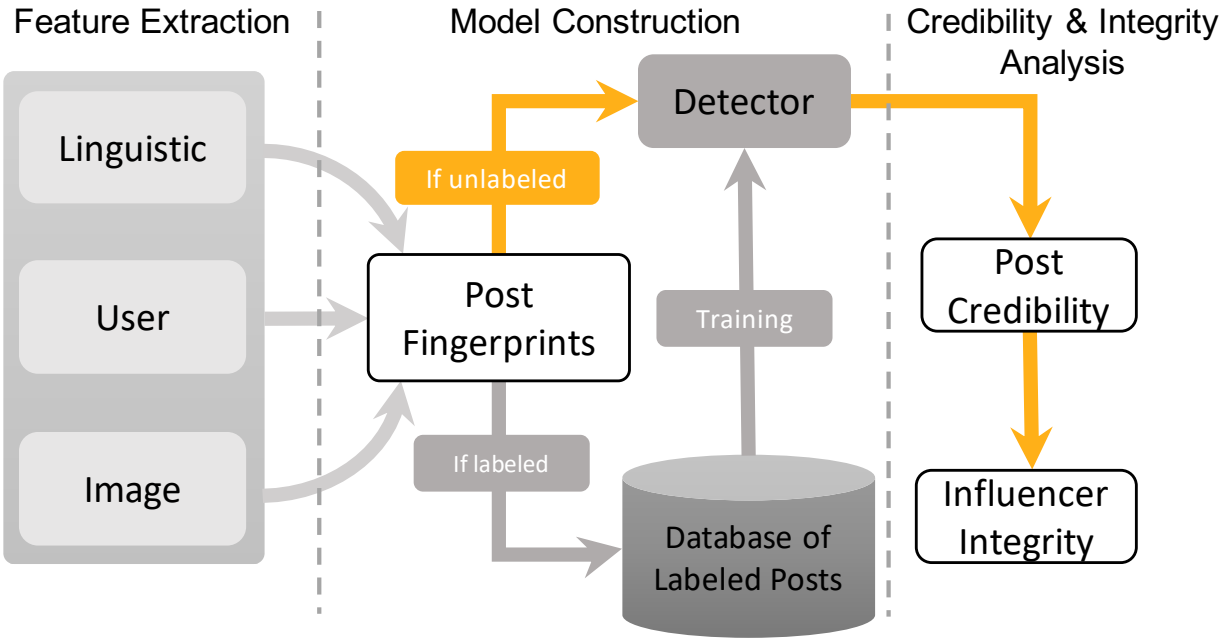

Figure 5.2: An overview of the USPD system

5.3.2 USPD Framework

Fig. 5.2 shows the overview of the USPD framework. It consists of three main phases: (1) feature extraction, (2) model construction and (3) credibility and integrity analysis.

1. The feature extraction phase selects features that can represent the content and related information of the post, which is called post fingerprints.

2. The model construction phase builds a machine learning detector to differentiate sponsored posts from unsponsored posts based on the selected feature representations for labeled posts. The collected posts will be automatically labeled if they are DSPs. For the training purposes, some non-DSP posts will be manually labeled as USP or unsponsored.

Table 5.1: Main symbols for sponsorship disclosure

symbols	definitions
DSP	disclosed sponsored post
USP	undisclosed sponsored post
s	an Instagram post
p	the publisher of the post
$\mathbf{S}$	social engagement for all posts
S_p	the historical posts of the publisher p
S_p^u	the corresponding n-th posts of the publisher p

3. The credibility and integrity analysis phase takes an unlabeled non-DSP post as the input to predict its credibility, i.e., whether it is a USP or an unsponsored post. We utilize the historical posts of an influencer to build an estimator for the analysis of the undisclosed sponsored posts ratio. Influencer integrity is then analyzed to evaluate the degree to which the influencer's posts can be trusted.

5.3.3 Feature Extraction

On Instagram, a post can have caption, image and user profile, which provide more information than a piece of standalone text. The intuition is that there are measurable differences between sponsored posts and unsponsored posts. In order to capture these differences, we analyze the characteristics of sponsored posts. It makes sense that these features would related to either the caption, the user and the image. Using this insight, we classify the post features into three categories: linguistic based features, user based features and image based

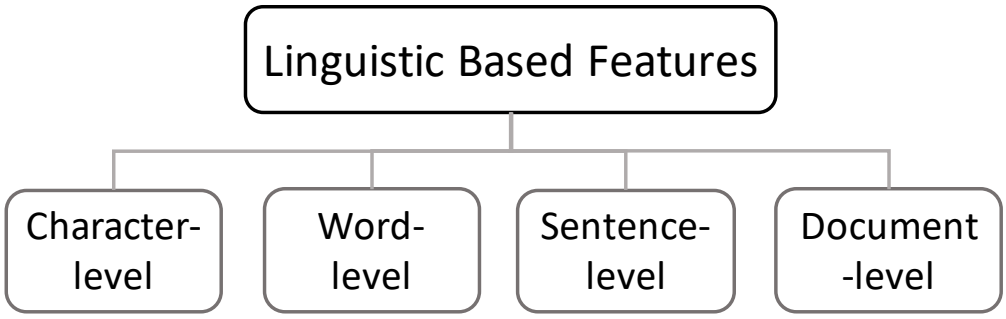

Figure 5.3: Level by level structure

features. Inspired by the classic text mining and image classification [50, 51], we create a list of interesting features for each of these three categories. The contribution of each features is studied to ensure it can help predict the sponsored posts.

5.3.3.1 Linguistic Based Features

In an Instagram post, the caption usually contains both text and emoticons. The linguistic features in the text can capture some characteristics of the caption of the post. Since sponsored posts are created for advertising purpose rather than describing something objectively, they often contain words of compliment and commendation. Thus, USPD framework exploits the linguistic features that capture the opinions and emotions in the caption text. To extract these features, the text content is separated into different levels, such as characters, words, sentences and documents. Accordingly, we explore the linguistic features level by level as shown in Fig. 5.3.

Character-level

We only have one feature, "Presence of emoticons", in character-level. It checks the presence of emoticons in the text by using the emoticon online dictionaries[52] and python3 library called *Emoji*.

Word-level

In word-level, we select six features, which consider the purpose and meaning for each word. They are shown in Table 5.2.

Sentence-level

Sentence complexity is the only feature in sentence-level. It denotes the depth of the dependency parse tree of the text. Usually, a commercial post is more formal and has more complex sentence structure than a personal post. The dependency parse tree for the sentence "won two back to back CSGOLotto games today on stream" in Fig. 5.1 is shown in Fig. 5.4. It has the following typed dependencies:

- won (VBD: verb, past tense)
- two (CD: cardinal number)
- back (RB: adverb)
- CSGOLotto (JJ: adjective)
- games (NNS: noun, plural)

Table 5.2: Features in word-level

Name	Description
Ratio of abbreviations	This feature is calculated by the ratio of abbreviations (e.g., UI for User interface) versus complete words, where the abbreviations can be checked via online libraries. Usually the publishers use fewer abbreviations than their daily emotion expressing posts.
Average word length	This feature counts the average number of characters in each word of the text, for example: "I like this clothing" contains 1, 4, 4, 8 characters respectively, with a total of 4 words, and the average number of characters is $(1+4+4+8)/4 = 4.25$. The intuition is that brands or merchants prefer to use longer and more complicated words than ordinary people.
Presence of opinion words	This feature checks the presence of opinion words such as "think", "believe" in the text. It is a binary feature.
Number of unique words	This feature calculate the total number of unique words in a caption text
Number of positive emotions	This features counts the number of positive emotions in the text, where emotion words can be checked via Linguistic Inquiry and Word Count (LIWC) [53]. Usually the publishers would add positive emotions like "heart", "smile".
Presence of @ words	This feature checks the presence of the words after @ in the hashtag list. The intuition is that for the caption of sponsored posts, the publishers usually '@' the brand or product key words.

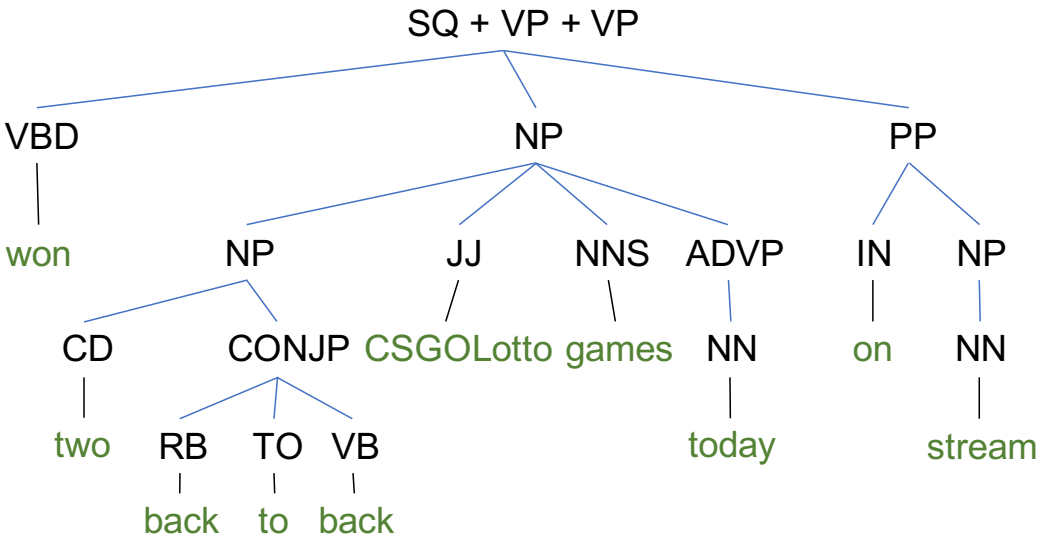

Figure 5.4: The dependency tree of the caption in Fig. 5.1.

- today (NN: noun, singular or mass)
- on (IN: preposition or subordinating conjunction)
- stream (NN: noun, singular or mass)

The abbreviations such as VBD, NN are used to represent the part-of-speech tags. Then, we can recover phrases constructed by the part-of-speech tags via chunking. The list of tags can be found in [54]. In this example, the post, "Won two back to back CSGOLotto games today on stream.", has a depth of 5. Since Instagram caption is noisy and unconventional, the performance of NLP parsers could be constrained. This is an area that can be explored further in the future to improve our detection system.

Document-level

For document-level, it consists of 3 features, which evaluate the meaning as a paragraph level as shown in Table 5.3.

Table 5.3: Features in document-level

Name	Description
Presence of external links	This feature checks the presence of external links in the text. Sponsored posts are likely to contain external links for more details about the products or to help place the order easier.
Polarity of the text	This feature measures the ratio of positive, neutral, and negative words in the caption text. Usually the publishers would add more positive words to attract customers' attentions and interests.
Viewpoint of the text	This feature measures users' opinion towards the whole text, such as supporting and denying [55]. It is an opinion mining feature.

5.3.3.2 User Based Features

To make the advertisements more effective, companies usually sponsor celebrities as they have higher influence. More important users are more likely hired by companies. Meanwhile, more influential celebrities may be more trustful as they may not want to ruin their reputation. This situation brings us the opportunity to investigate the credibility of influencers. To identify the importance of users, we extract user based features. Specifically, we extract individual based and community based features.

Individual based

Celebrities generally have more followers and fewer followings. Therefore, we adopt the following three features:

- Number of followers: this feature counts the number of followers of a publisher. The more followers a publisher has, the more influential it is.
- Ratio of followers to followings: this feature measures the ratio of followers to followings of a publisher. It presents the role of the publisher in the network. If a publisher has low follower to following ratio, we assume that he is a receiver.
- Verified account or not: this feature measures whether a user account is verified or not. It is a binary feature.

Community based

Important (or high influence) users likely have more followers who are also important. PageRank [12] exactly models this intuition. The matrix form of PageRank we use is as follows:

$$\mathbf{r} = c\mathbf{P}^\top \mathbf{r} + (1-c)\mathbf{1}/n \tag{5.1}$$

where $\mathbf{r}$ is the PageRank vector with $\mathbf{r}_i$ representing importance of user i, followees(i) represents the number of followees of user i, $\mathbf{P}$ is the node-to-node transition matrix with $\mathbf{P}_{i,j} = 1/\text{followees}(i)$ representing the transition probability from user i to j, $\mathbf{P}^\top$ represents the transpose of matrix $\mathbf{P}$, and $\mathbf{1}$ is a vector of all 1's. Since PageRank values are mainly de-

termined by the community a user belongs to, we use the PageRank values as the community based features.

5.3.3.3 Image Based Features

Pictures along with text may convey more information to the audience than text only. They can convey emotions, ideas, thoughts and reality, which are difficult to achieve through pure text. They also bring the audience into the world of the publisher, which can better resonate with the audience. However, it is generally infeasible to integrate all of image information in a reasonable running time. Thus, it makes sense to extract image based features that captures image characteristics.

Object detector

Sponsored posts usually have fewer objects in the post images. The reason is that the advertisers do not want too many objects to distract people's attention. In Computer Vision, face detection, pedestrian detection and instances of certain classes like buildings and cars are well-researched. Generally, object detection can be achieved by machine learning or deep learning approaches. Histogram of Oriented Gradients (HoG) [56] feature descriptor combined with support vector machine (SVM) is popular for object detection. The technique counts the occurrences of gradient orientation in localized portions of an image. For deep learning approaches, R-CNN [57] is one of the first deep learning based object detectors, which is the basis for methods such as Fast R-CNN [58] and Faster R-CNN [59]. Here we use YOLOv3 Object detector [60] to count the number of objects in the image.

Logo text detector

Sponsored posts usually contain company logos in the caption and image. Hence we use optical character recognition (OCR) to recognize texts in the image. Furthermore, we search the detected texts in the caption. The feature is set to one if the caption contains the text and zero otherwise. Tesseract OCR library is employed to utilize a Long Short-Term Memory (LSTM) network for high accuracy in text recognition.

5.3.4 Model Construction

The USPD framework adopts an ensemble machine learning classifier, which ensembles the Logistic Regression, Support Vector Machine and Random Forest by selecting the best performance with the majority voting.

Logistic Regression (LR) uses a logistic function, which is also called sigmoid function, to model a binary classification problem. The curve of this function can take any real number and map it into a range between 0 and 1. The LR equation is as the following:

$$y = \frac{1}{1 + e^{-(\mathbf{w} \cdot x)}} \tag{5.2}$$

where y is the predicted probability of an instance belonging to a class, $\mathbf{w}$ are the parameters of the model.

Support Vector Machine (SVM) is a discriminative classifier defined by separating hyperplane. Based on the given labeled training data, SVM generates an optimal hyperplane which can categorize a new input. The intuition of SVM is to find a separating hyperplane

such that the distance between support vectors and the separating hyperplane is maximized. Different with LR, which considers all points to the separators, SVM only considers points at the boundary. The loss function for SVM can be transformed with Lagrange duality [61, 62] as:

$$\text{minimize } L(\lambda) = \frac{1}{2}\sum_{i,j=1}^{n} \lambda_i \lambda_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j - \sum_{i=1}^{n} \lambda_i \tag{5.3}$$

$$\begin{aligned} \text{subject to} : \sum_{i=1}^{n} \lambda_i y_i = 0 \\ \lambda_i \geq 0 \end{aligned} \tag{5.4}$$

where y_i is the label for data $\mathbf{x}_i$, λ_i is the variable brought by Karush-Kuhn-Tucker theorem [63]. If the i^{th} data is not support vector, λ_i will equal to zero, otherwise, larger than zero. The kernel of the SVM can determine the shape of the separating hyperplane. In Eq. (5.3), the kernel is $\mathbf{x}_i \cdot \mathbf{x}_j$, which is a linear kernel. The regularization parameter is the C parameter in the scikit-learn library, which controls the margin of the separating hyperplane.

Random Forest (RF), as an ensemble machine learning algorithm, builds multiple decision trees and merges them together. It adds an additional layer of randomness to bagging. In standard trees, each node uses all variables to choose best split. In a random forest, each node is split using the best among a subset of predictors randomly chosen at that node. This strategy turns out to perform very well compared to many other classifiers, including support vector machines and neural networks, and is robust against overfitting [64]. RF does not need to tune hyper-parameter, but searches the best result among trees with random subsets of features.

5.3.5 Credibility and Integrity Analysis

Once the model is constructed, we use it to predict whether an incoming post is a USP or not. We check influencers' historical posts and calculate the ratio of USPs as in Eq. (5.5) to denote the integrity of the influencers. Let p be a publisher, $\mathbf{S}_p^{USP}$ be the set of all USPs of p, $\mathbf{S}_p^{DSP}$ be the set of all DSPs.

$$\text{Integrity}(p) = \begin{cases} 1, & \text{if both } |\mathbf{S}_p^{USP}| \ \& \ |\mathbf{S}_p^{DSP}| = 0, \\ 1 - \dfrac{|\mathbf{S}_p^{USP}|}{|\mathbf{S}_p^{USP}| + |\mathbf{S}_p^{DSP}|}, & \text{otherwise.} \end{cases} \tag{5.5}$$

where $|\mathbf{S}_p^{USP}|$ and $|\mathbf{S}_p^{DSP}|$ are the cardinal numbers of $\mathbf{S}_p^{USP}$ and $\mathbf{S}_p^{DSP}$, respectively.

The higher integrity value for an influencer, the more trustable their posts will be. For example, if an influencer has 5 DSPs and 5 USPs, the integrity value for it will be 0.5. If an influencer has purely DSPs or purely USPs, the integrity value for it will be 1 or 0, respectively. This way, the formula will capture all the historical posts for influencers that mix disclosed posts with undisclosed sponsored posts (to avoid FTC detection).

5.4 Performance Evaluation and Discussion

In this section, we present our experimental results and compare the performance of our framework with the ground truth datasets.

5.4.1 Dataset

we evaluate the accuracy of our USPD framework on the Instagram dataset. Instagram posts are scraped to create the whole training and testing datasets. First, we crawl posts with "#sponsored" and "#ad" hashtags. Since Instagram restricts the number of retrieved posts during each login access, we ran the crawl program twice a week in January 2019 with a total number of 6 times. After removing the duplicate and non-English posts, we obtain 43,738 DSPs in total. Then, we randomly crawl 63,819 posts as non-DSP posts, which are labeled by experienced Instagram users as USP or not. After this annotation phase, we have 7,314 USPs and 56,505 unsponsored posts (UPs) out of all non-DSP posts. Table 5.4 summarizes the dataset.

To analyze the integrity of influencers, we select 1,000 publishers with more than 10,000 followers. Then, we crawled their posts within recent four years or 1,000 historical posts from these chosen publishers, whichever is lower.

Table 5.4: Statistics of collected Instagram posts

Classes	Number of posts
Disclosed Sponsored Posts (DSPs)	43,738
Undisclosed Sponsored Posts (USPs)	7,314
Unsponsored Posts (UPs)	56,505
Total	107,557

5.4.2 Data Preprocessing and Experimental Setup

The raw data was transformed from the original format into a new dataset consisting of 21 features as described in Section 5.3. The data is then normalized into the range of [0, 1] by using the Min-Max normalization in Eq. (5.6):

$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)} \tag{5.6}$$

where z_i is the normalized data for x_i in $x = (x_1, ..., x_i, ..., x_n)$.

We use Chi-Square test to measure the contribution of each feature. The null hypothesis is that there is no significant association between a feature and the outcome after taking other features into account in the model. Table 5.5 shows the significance of features determined from the Chi-Square test at 0.05-level ($p < 0.05$) with total number of 15 features.

5.4.3 Evaluation Metrics

To evaluate the performance of the framework and compare the results with ground truth, we collect the following numbers.

- True Positive (TP): the number of posts predicted sponsored that are actually annotated as sponsored posts;
- False Positive (FP): the number of posts predicted sponsored that are actually annotated as unsponsored posts;
- False Negative (FN): the number of posts predicted unsponsored that are actually annotated as sponsored posts;

Table 5.5: Top 15 statistically significant features with Chi-Square test

Verified account or not
PageRank value
Presence of external links
Average word length
Number of followers
Presence of opinion words
Ratio of abbreviations
Number of positive emotions
Presence of @ words
Sentence complexity
Polarity of the text
Viewpoint of the text
Ratio of followers to followings of a user
Object detector
Logo text detected

- True Negative (TN): the number of posts predicted unsponsored that are actually annotated as unsponsored posts;

The following metrics are then defined.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{5.7}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{5.8}$$

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{5.9}$$

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5.10}$$

These metrics allow us to evaluate and compare the performance of models quantitatively from different aspects. Accuracy measures the overall performance of the framework with real-world data. Since Instagram posts may be skewed, the accuracy value can be high by making more TN predictions. Thus, we also consider the precision value which measures the fraction between the number of true sponsored posts and the number of predicted sponsored posts. F1 score is the combination of precision and recall. It can provide a general performance evaluation for the detection framework. The higher value of these four metrics are, the better performance of the framework will be.

To return a less biased estimation of the model, we use ten-fold cross validation. We shuffle the USPs and UPs and equally split the dataset into ten parts. We develop two models for our experiments: model_A and model_B. For model_A, we use all DSPs, the split USPs and UPs to train the model, the rest one part of USPs and UPs for testing. For model_B, we use one part of split data as testing data and the rest as training data. For each model, to reduce the cost function, we tuned all parameters to achieve high classification

Table 5.6: Performance results of classifiers

Classifier	**Accuracy**	**Precision**	**Recall**	**F1**
RP(model_B)	0.52	0.36	0.49	0.42
RP(model_A)	0.47	0.42	0.58	0.48
LR(model_B)	0.76	0.30	0.84	0.44
LR(model_A)	0.71	0.47	0.79	0.59
SVM(model_B)	0.77	0.41	0.89	0.56
SVM(model_A)	0.78	0.69	0.84	0.75
RF(model_B)	0.79	0.44	0.84	0.57
RF(model_A)	0.82	0.72	0.86	0.78
Our Model(model_B)	0.79	0.62	0.85	0.72
Our Model(model_A)	0.83	0.74	0.88	0.80

accuracy.

5.4.4 Results and Discussion

Table 5.6 shows the comparison between our proposed ensemble model and traditional models such as random predictor (RP), Logistic Regression (LR), Support Vector Machine (SVM) and Random Forest (RF). From this table, we can see that our proposed model_A achieves the best performance in terms of accuracy, precision, recall and F1 score. This is because our model utilize the best performance for each basic model and make the final decision by majority vote. The accuracy difference for model_A and model_B is less than that of F1

score. This is because accuracy takes true negative (TN) into consideration. In our case, the size of USPs is much smaller than UPs, the classifier does not have enough data to train the model, which leads to a high TN value. In the meantime, the F1 score for model_A is relatively high, which shows that the disclosed and undisclosed sponsored posts do share some similarities and using DSPs to train the model can improve the model performance. Thus, we choose model_A to further evaluate the framework performance. Meanwhile, Random Forest achieves higher accuracy than the RP, LR and SVM. Because it creates decision trees on randomly selected data samples, which in return helps to make the model generalize better. Note that Random Forest algorithm also provides the feature importance value. By using the scikit-learn library, we find the top three important features are logo text detected, number of positive emotions and the presence of @ words in hashtag lists.

Since our model performs the best with an overall accuracy of 0.83, from this point on, the model being discussed is our ensemble model. To measure the contribution for each categories, we evaluate the accuracy for only one category of features as is shown in Fig. 5.5. The shadow bars show the accuracy of our framework using each of the three categories of features independently. We can find that the model trained on the linguistic based features and image based features outperforms the user models by a sizeable margin. When we remove only one category of features, we can check the influence of each category of features as shown in Fig. 5.6. The white bars show the accuracy of the framework without one category of features, and the shadow bars show the accuracy decrease after removing this category of features. The higher shadow bar is, the more important the category of features

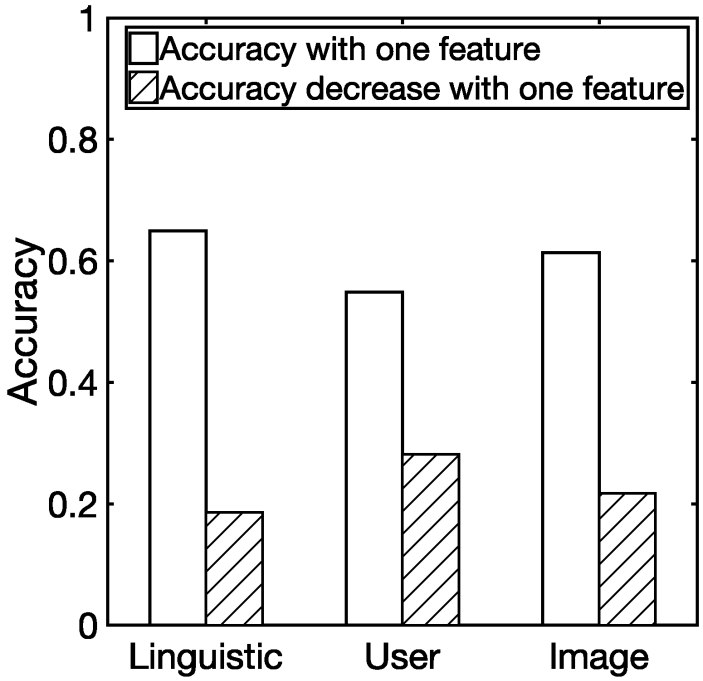

Figure 5.5: The accuracy with only one category of features and their accuracy decrease compared with three categories of features

is. From Fig. 5.6, we can see that both linguistic and image categories have relatively high importance, while user based category is less important.

To study the integrity of the influencer, we rank the influencers based on the number of followers and select top 1,000 influencers to calculate the probability as shown in Eq. (5.5). Our results show that the percentage of the top 1,000 influencers with 0.6 integrity probability is higher than 70%.

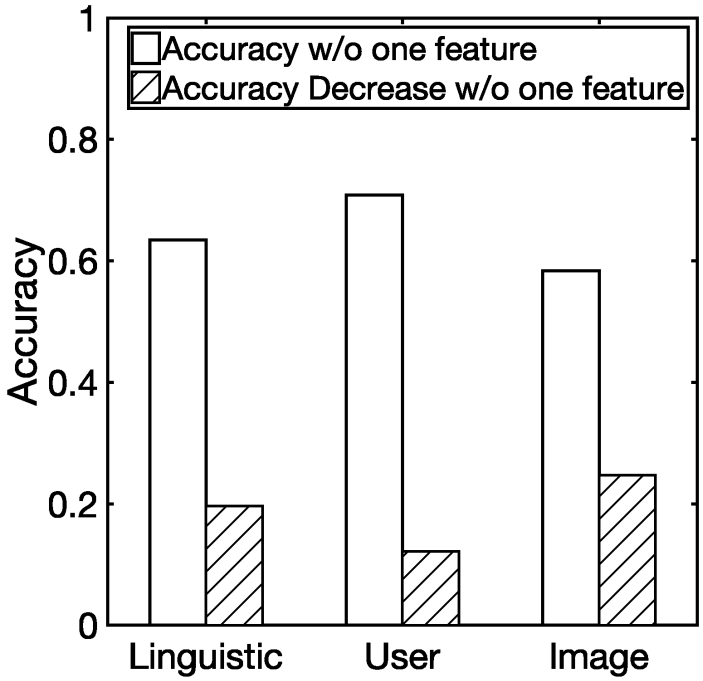

Figure 5.6: The accuracy with only two categories of features and their accuracy decrease compared with three categories of features

5.5 Chapter Summary

With the wide spread of machine-to-machine communication, the interaction between people through smart devices becomes challenging in terms of trust. It is hard to know the people you are communicating with from the other side of the internet is trustworthy or not. Instagram is one popular online social networking platforms and the emphasis of visuals makes it stand out for people believing in "To see is to believe". The large user quantity makes Instagram a critical place for business marketing campaigns. Brands and merchants pay Instagram influencers for their product advertising posts. Some influencers do not disclose their partnership with the brands, which may give a misleading impression to the followers.

In this work, we have developed an effective Undisclosed Sponsored Post Detection (USPD) framework to automatically detect if a post is an undisclosed sponsored post or not. The proposed framework takes advantages of the text, user and image features in the posts to analyze the integrity of an influencer, or how much an influencer can be trusted. Based on our extensive experiments, we find the proposed feature categories significantly better than the random predictor. This indicates that as the first work in this domain, the proposed features are very powerful in Instagram post analysis. In the next step, we will try to extend the proposed feature categories to fusion features. In addition, we should note that the proposed model utilize some current state-of-the-art libraries package to achieve feature extraction. However, these libraries do have their limitation and may not apply to Instagram specific domain knowledge. In future work, these features should be considered.

Since our work currently is focused on English-language Instagram, it would be feasible to extend our work to handle posts in other languages. The challenging part would be analyzing the caption structure for the non-English language and extracting the image information. Meanwhile, similar techniques could potentially be applied to other online and publicly available social media platforms (e.g. Twitter). Some of the features are platform-agnostic and can be extracted and processed for different platforms directly. Besides, in addiction to predict the sponsorship of the post, we could further predict the impact of these sponsored posts would have on individuals and communities, such as the total number of people being exposed to the advertisement posts, the reaction after being exposed.

CHAPTER 6
TRIPARTITE GRAPH CLUSTERING IN ONLINE SOCIAL NETWORKS

The COVID-19 pandemic has hit the world hard. The reaction to the pandemic related issues has been pouring into social platforms such as Twitter. Many public officials and governments use Twitter to make policy announcements. People keep close track of the related information and express their concerns about the policies on Twitter. It is beneficial yet challenging to derive important information or knowledge out of such Twitter data. In this chapter, we collect the tweets containing a set of keywords related to coronavirus pandemic as the ground truth data and build a Tripartite Graph Clustering for Pandemic Data Analysis (TGC-PDA) framework to automatically detect the communities of Twitter users and analyze the topics that are discussed in the communities. Such information would provide policy makers useful information and statistics for decision making.

6.1 Introduction

The coronavirus (COVID-19) has hit the world and made major impacts in society, economy, medical care, environment, and so on [1, 65]. To prevent the virus from spreading all over the world, unnecessary travel has been restricted, quarantines are required and even Tokyo Olympics is postponed [66, 67, 68]. Fig. 6.1 is the map of the verified number of COVID-19 infections as of 09/19/2020 [1]. The deeper color represents a higher number of cases. The grey color means no reported cases, or no data available at the time. We can see that almost the whole world is non-gray and the virus is widespread. Since the major pathogen of COVID-19 is a single-stranded RNA, it is intrinsically unstable. The immune system of

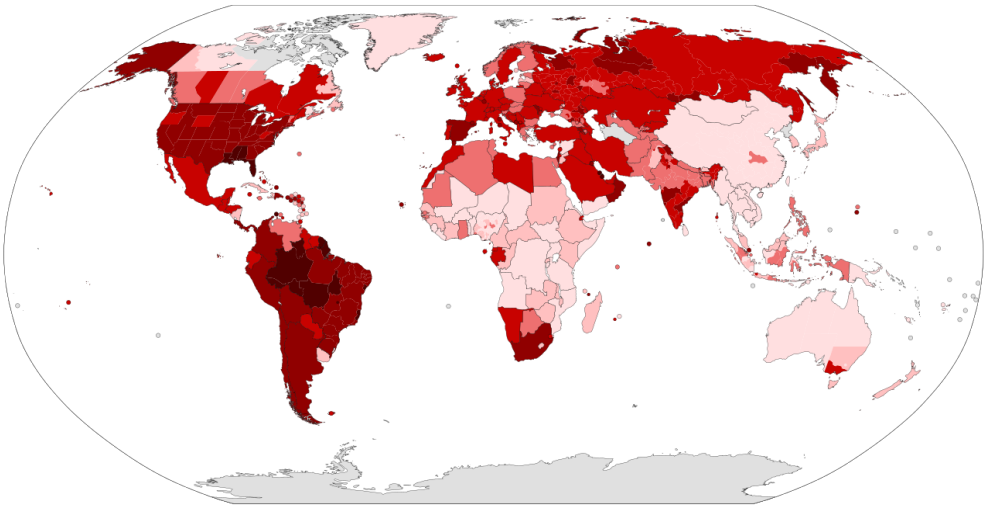

Figure 6.1: The COVID-19 outbreak world map per capital [1].

humans may not be as responsive (even injected with vaccine) if the virus RNA significantly changes its structure [69]. This means the vaccine may need to be constantly updated and there could be a lengthy recovery for many people.

Meanwhile, the pandemic has led to the global economic disruption, the anxiousness for supply shortages and the fear of disease. A mass of misinformation and conspiracy theories about the coronavirus have spread over the Internet, especially on Online Social Networks (OSNs) [70]. Twitter, as one of the most popular social networking platforms, is widely used to allow people to post and comment the messages called "tweets". Twitter contains not only the text data, but also the interaction among users. The ever-increasing expansion of the Internet and mobile networks allows real-time propagation of tweets to a large number of people, which makes it a desired environment for the breaking-news discussion and fear contagion. The open source intelligence from the social networks in Twitter can be utilized to analyze and keep track of the attitudes or opinions of people towards coronavirus pandemic

events. Important information or knowledge can be derived from such Twitter data, which can provide policy makers better information and statistics for decision making.

In Twitter, the data (such as users and tweets) are linked rather than a bunch of standalone information units. Thus, it is natural to represent such linked data as graphs. The representation of the graph can largely affect the performance of Twitter data analysis. Meanwhile, the scale of the graphs increases explosively from thousands of vertices to billions of vertices, which makes it important to find a proper graph representation for the data. A suitable graph representation can make the entire descriptive or predictive graph analyzing process efficient and effective. For example, multipartite graphs can be used to model networks with different objects, such as documents and terms, movies and preferences, or buyers and sellers. In [71, 72], unipartite and bipartite graphs have been used to represent Twitter data. Once we have a proper graph presentation of Twitter data, machine learning models can be developed to conduct various data analysis including sentiment, prediction, trend analysis and so on [73, 74].

Sentiment analysis in Twitter can analyze the tweet texts to identify the opinions or ideas that users express. Much literal work on Twitter sentiment analysis focused on understanding the sentiments of individual tweets and user-level sentiments [75, 76, 77]. Some researchers studied both tweet-level and user-level sentiments [78, 79]. Sentiment analysis is challenging because the sentiments of users are correlated with the sentiments expressed in many short tweets, which are intrinsically noisy and labile. In addition, it is difficult to understand and characterize the dynamics in user's sentiments, as different time may lead to contradict

opinions towards the same topic. It is not uncommon to see people having a lukewarm and reluctant attitude towards a product at first glance, but later cannot live without it.

In this work, we investigate the issue of pandemic analysis through Twitter data, and propose a framework of Tripartite Graph Clustering for Pandemic Data Analysis (TGC-PDA). In the proposed framework, we first build a tripartite graph, which will take advantage of the characteristics of the Twitter users and tweets network structure to facilitate exploring the community structures. Then we cluster the tweets and users based on the structure of the graph using a clustering approach. Finally, the framework provides the open source intelligence from the clustered communities, which can help keep track of people's feedbacks and opinions towards the Coronavirus pandemic events. To the best of our knowledge, the TGC-PDA framework is the first to effectively analyze the sentiments of users for COVID-19-related topics based on transforming of the tripartite graph. We conduct a set of experiments on COVID-19 related Twitter data, and verify that our approach is effective and efficient.

The rest of the chapter is organized as follows. Section 6.2 introduces the background of our work. Section 6.3 explains the main notations and problem formulation for our work. Section 6.4 explains our proposed framework. Section 6.5 illustrates the computing algorithm. Section 6.6 elaborates the experimental setting, results and analysis. Finally, conclusions are drawn in Section 6.7.

6.2 Background

In this section, we present some backgrounds on multipartite graph modelling, non-negative matrix factorization and sentiment analysis.

6.2.1 Multipartite Graph and Community Detection

A multipartite graph is usually used to model heterogeneous data. In graph theory, a multipartite graph or k-partite graph is a graph whose vertices are or can be partitioned into k different independent sets. Equivalently, it is a graph that can be colored with k colors, while no two endpoints of an edge have the same color. Fig. 6.2 shows two examples of multipartite graph with different node sets. When $k = 2$, a k-partite graph is called a bipartite graph. When $k = 3$, a k-partite graph is called a tripartite graph, which generally means it is constructed using data from three heterogeneous sources.

Fig. 6.3 shows an example of clustering results for a tripartite graph. The solid lines are the edges connecting different types of nodes. The dashed line polygons cluster the nodes into two different groups, which form two communities. Clustering is an unsupervised approach, no labelled data is required. Compared with the traditional clustering approaches built on just the information from a graph with one type of node, clustering multipartite graph nodes (with different types of nodes) can derive more meaningful and useful statistics or information from the graph.

For unipartite graphs, there are usually two main approaches to detect communities. One is based on modeling the community structure or topology and the other is based on extract-

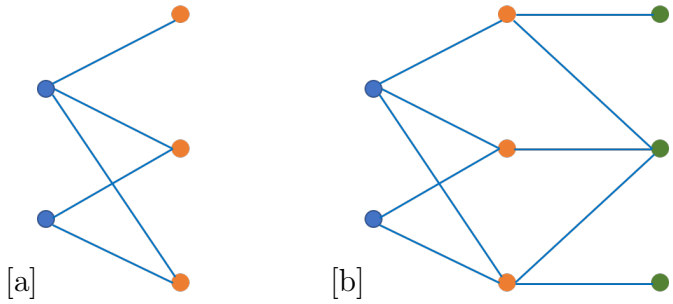

Figure 6.2: Examples of multipartite graph with different k: [a] $k = 2$, [b] $k = 3$.

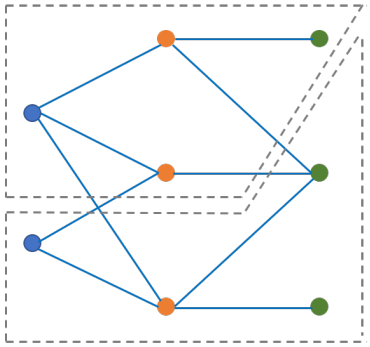

Figure 6.3: An example of tripartite graph co-clustering problem.

ing it from flow calculations. Analyzing multipartite graph as a separate network category to investigate the community structure has become popular in the literature [80, 81]. To handle the multipartite graph cases, one popular way is to utilize the edge features to simplify the multipartite graph, as edges usually are one type. However, such transformation may loss too much information to determine communities accurately. Recently, researchers have proposed approaches based on non-negative matrix factorization (NMF) [82] and ranking [83].

6.2.2 Non-negative Matrix Factorization (NMF)

NMF is a technique for obtaining low rank representation of matrices that have only non-negative elements [84]. It has many applications including information retrieval and text

mining [85]. The constraints of non-negative basis vectors differ from other rank reduction method such as principal component analysis (PCA) [86] and singular value decomposition (SVD) [87], which makes it ideal for situations where non-negative numbers are used for data interpretation/representation, for example, pixel intensities in image processing. Generally, NMF aims to factor a data matrix $\mathbf{A}$ into two lower dimension matrices ($\mathbf{W}$, and $\mathbf{H}$) and minimize the square error between the original matrix $\mathbf{A}$ and the multiplication of those two matrices as shown in Eq. (6.1):

$$\begin{gathered} \min ||\mathbf{A} - \mathbf{WH}||_F^2 \\ \text{subject to } \forall i, j, [\mathbf{W}]_{i,j}, [\mathbf{H}]_{i,j} \geq 0 \end{gathered} \tag{6.1}$$

In Eq. (6.1), $\mathbf{W}$ and $\mathbf{H}$ are called dictionary matrix and expansion matrix. The challenge here is how to effectively identify these two matrices. There are multiple variations to the basic NMF approach wherein additional constraints such as sparsity and orthogonality, are imposed to limit the solution space for the decomposed result. Decomposing into three matrices has also been proposed. It is called non-negative matrix tri-factorizations and it shows good performance in approximation [88].

The optimization problem in Eq. (6.1) is a non-convex optimization with respect to variables $\mathbf{W}$ and $\mathbf{H}$. Thus, a local optima is often encountered. To find optimized solutions for the matrices, one common approach to update matrices is the multiplicative updating algorithm [84]. However, it has poor performance [89] and convergence issues[90]. There are some other approaches such as the block principal pivoting method [91] and the random projections method [92] that can be used to update matrices. These methods try to over-

come the shortcomings of the multiplicative updating algorithm, and generally have better performance.

6.2.3 Sentiment Analysis

Sentiment analysis or opinion mining is a natural language processing technique used to extract subjective information, usually in three classes: positive, neutral and negative. A large number of research in sentiment analysis focused on identifying text polarity [93, 94, 95, 96, 97]. There are some other research focusing on feelings and emotions such as angry or happy, and intentions [98, 99, 100]. Sentiment analysis can help gauge public opinion, conduct nuanced research and monitor large data tends effectively.

6.3 Pandemic Analysis through Twitter Data

In this section, we discuss how we construct the tripartite graph, the notations and the problem formulation for pandemic analysis.

6.3.1 Tripartite Graph in Twitter

Th important information of a tweet includes: (i) user, (ii) tweet text, and (iii) hashtag/keyword. The relationships among them are straightforward: a user can like or comment or post a tweet, and a tweet might have some topics/hashtags/keywords. In other words, users will perform actions (e.g., like/comment) on tweets, while each tweet is associated with certain topics/hashtags/keywords. As users do not directly perform actions on the topics/keywords/hashtags, we can abstract the relationships among user, tweet contents, and

topics/keywords/hashtags into a tripartite graph. For example, Fig. 6.4 is an example to model Twitter data as a tripartite graph. The tripartite graph is composed of three types of nodes: user nodes, tweet nodes, and topic nodes. In the tripartite graph, the user nodes only connect with the tweet nodes, while the tweet nodes only connect with the topic nodes. In Fig. 6.4, the solid lines with red heart icon and message icon represent like or comment relationship between users and tweets, respectively; while the lines without icon represent the containing relationship between tweets and topics.

6.3.2 Problem Definition

We denote the raw data from the Twitter platform with a 3-tuple $RD =< \mathbb{U}, \mathbb{T}, \mathbb{H} >$, where $\mathbb{U}, \mathbb{T}, \mathbb{H}$ represents the set of users, tweets and topics, respectively. Note that $\mathbb{U}$, $\mathbb{T}$, and $\mathbb{H}$ are three mutually disjoint sets (i.e., $(\mathbb{U} \cap \mathbb{T}) \cup (\mathbb{T} \cap \mathbb{H}) \cup (\mathbb{H} \cap \mathbb{U}) = \emptyset$).

Given the raw data, our target is to generate community's attitude towards COVID-19 events via the following phases: (i) generate tripartite graph representation from the raw

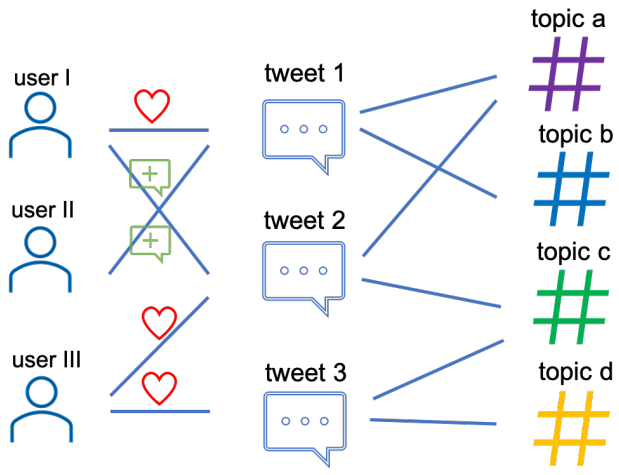

Figure 6.4: An example of tripartite graph in Twitter

Table 6.1: Notations for TGC-PDA

symbols	definitions
n, m, t	number of users, tweets and topics
$G(V, E)$	graph with node set V and edge set E
$\mathbb{U}, \mathbb{T}, \mathbb{H}$	Node set of users, tweets and topics
$\mathbf{B}$	matrix representation of a bipartite graph
$P_{i,j}$	number of paths between node i and node j
$\mathbf{L}$	normalized Laplacian matrix, $\mathbf{L}=\mathbf{I}-\mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}$
$\mathbf{D}$	degree matrix: diagonal with $[\mathbf{D}]_{i,i} = degree(v_i)$
$\mathbf{F}, \mathbf{G}$	decomposed matrices: $\mathbf{F} \in \Psi^{n\times d}$ and $\mathbf{G} \in \Psi^{k\times n}$
$\mathbf{S}$	association matrix: $\mathbf{S} \in R_+^{d\times k}$
Ψ	the set of all cluster indicator matrices
$\mathrm{Tr}(\mathbf{X})$	trace of matrix $\mathbf{X}$: $\mathrm{Tr}(\mathbf{X}) = \sum_1^n x_{i,i}$
$\|\|\mathbf{X}\|\|_F$	Frobenius norm of a matrix $\mathbf{X}$

data, (ii) detect the communities via graph clustering, and (iii) infer sentiments for each community. The attitude of each community will be represented by positive or neutral or negative. Table 6.1 shows the notations used in this work.

6.4 Pandemic Data Analysis Framework

In this section, we propose a framework of Tripartite Graph Clustering for Pandemic Data Analysis (TGC-PDA), to automatically collect, cluster and infer the sentiments from the observed tweets.

Fig. 6.5 shows the overview of the TGC-PDA framework. The input of the framework is

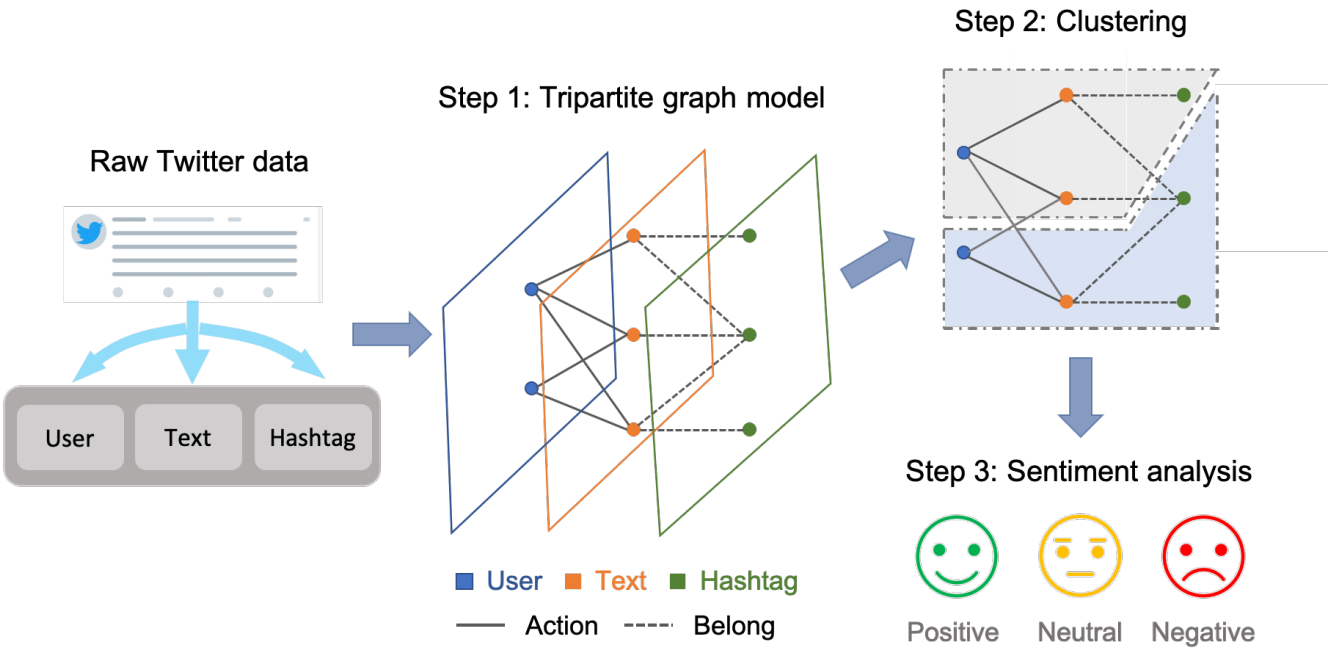

Figure 6.5: An overview of the TGC-PDA framework

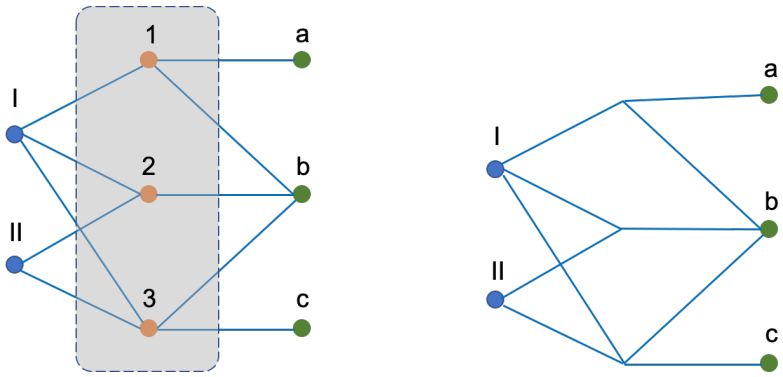

Figure 6.6: Build the user-topic bipartite by removing the tweet nodes of the tripartite graph, and leveraging the tweet nodes as the connection for user and topic nodes.

Twitter raw data. TGC-PDA consists of three main steps: (i) tripartite graph representation, (ii) clustering and (iii) sentiment analysis. In the tripartite graph representation step, we find a mathematical model to represent the data with less information loss. Then, the clustering step builds a matrix factorization based clustering algorithm to find the communities in the

graph. Finally, the sentiment analysis step extracts attitudes within the communities.

6.4.1 Tripartite Graph Representation

A tripartite graph $G = (V, E)$ can be constructed from the data to represent the relationships among $\mathbb{U}$, $\mathbb{T}$, and $\mathbb{H}$, where V and E represent the node set and edge set, respectively. In G, we have $V = \{\mathbb{U} \cup \mathbb{T} \cup \mathbb{H}\}$, and $E = \{E_{\mathbb{U},\mathbb{T}} \cup E_{\mathbb{T},\mathbb{H}}\}$, while $E_{\mathbb{U},\mathbb{T}}$ is the edge set between the user nodes and the tweet nodes, and $E_{\mathbb{T},\mathbb{H}}$ is the edge set between the tweet nodes and the topic nodes. In graph theory, a tripartite graph is complete if and only if each node in one set of nodes is fully connected with all nodes in the adjacent set. Based on the data we obtained from Twitter, the tripartite graph generated in our case is not complete.

Different from the traditional machine learning techniques or frameworks [11, 101], the TGC-PDA will employ the proposed tripartite graph structure to organize the data. With the tripartite graph, the arising challenge is to identify a proper representation of the graph to embed useful information for further processes such as clustering and sentiment analyzing. To represent the graph and find a suitable clustering solution for a tripartite graph, one way is to divide the graph into two bipartite graphs. For example, we can build user-tweet bipartite graph and tweet-topic graph and then find clusters over these two graphs separately. In this case, the connections between user and topics are lost. In addition, the relations between users, which has been proved important in social network analysis, are not considered either.

Based on the analysis above, we propose to build a user-topic bipartite graph and a user-tweet bipartite graph, as shown in Fig. 6.6. We use tweet-level nodes as the bridges to build the connection between user and topic nodes. In in Fig. 6.6, node I has three paths

(a path is a finite sequence of edges connecting two end nodes) connecting to node b, which goes through node 1 and node 2. Accordingly, we can set the edge weight between node I and node b as 3. The user-topic bipartite graph can be denoted by $\mathbf{B_h}^{n\times t}$ and the matrix representation for user-tweet bipartite graph is denoted by $\mathbf{B_u}^{n\times m}$. Both of the bipartite graph matrices are non-negative and the detailed deduction will be discussed in Section 6.5.

6.4.2 Non-negative Matrix Factorization with Regularization

In the second step of the framework, we need to find the clustering result of the input graph data. Since the matrix representation of the graph is non-negative matrix, it is straightforward to use the Non-negative Matrix Factorization (NMF) for clustering. In this way, for the user-topic bipartite graph generated in Section 6.4. A, we can find an intermediate clustering result of the graph by applying the clustering algorithms. Then, we can feed the intermediate clustering result into clustering process of the user-tweet bipartite graph and the clusters can be found accordingly.

Because users tend to have consistent preferences, it would be preferable to make tweet nodes close to their user nodes. In other words, node locality needs to be preserved. Thus, standard NMF may not work properly. In fact, as to be described later, our experiment results show that the accuracy of NMF is poor. Hence, we propose to the graph regularization technique into NMF to smooth the result [102]. In specific, to cluster users and topics, we use the NMF with regularization (NMFR) modeled as the following equation:

$$\min\ ||\mathbf{B}_h\mathbf{B}^\top{}_h - \mathbf{F}_h\mathbf{S}_h\mathbf{G}_h||_F^2 + \alpha\,\mathrm{Tr}(\mathbf{F}_h^\top\mathbf{L}_f\mathbf{F}_h) + \beta\,\mathrm{Tr}(\mathbf{G}_h\mathbf{L}_g\mathbf{G}_h^\top) \tag{6.2}$$

where $||\mathbf{X}||_F$ denotes the Frobenius norm of a matrix $\mathbf{X}$. The Frobenius norm is a matrix norm of an $m \times n$ matrix $\mathbf{X}$, which is defined as the square root of the sum of the absolute squares of all elements as shown in Eq. (6.3):

$$||\mathbf{X}||_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|x_{i,j}|^2} \tag{6.3}$$

In Eq. (6.2), $\mathbf{B}_h$ is the matrix representation of the user and topic bipartite graph, which can be defined as Eq. (6.4). Here, $P_{i,j}$ is the number of paths between node i and node j.

$$[\mathbf{B}_h]_{i,j} = \begin{cases} P_{i,j} & \text{if } i \text{ and } j \text{ are connected} \\ 0 & \text{otherwise} \end{cases} \tag{6.4}$$

For example, the matrix representation of Fig. 6.6 will be:

$$\mathbf{B}_h = \begin{array}{c} \\ \mathsf{I} \\ \mathsf{II} \end{array} \begin{array}{c} \begin{array}{ccc} \mathsf{a} & \mathsf{b} & \mathsf{c} \end{array} \\ \begin{bmatrix} 1 & 3 & 1 \\ 0 & 2 & 1 \end{bmatrix} \end{array}$$

In Eq. (6.2), $\mathbf{B}_h\mathbf{B}_h^\top$ represents the pairwise similarity matrix, $\mathbf{F_h} \in \Psi^{n\times d}$ and $\mathbf{G_h} \in \Psi^{k\times n}$ are non-negative matrices and are cluster indicators, where Ψ is the set of all cluster indicator matrices. $\mathbf{S}_h \in R_+^{d\times k}$ is used to increase the degree of freedom such that the low-rank matrix representation has better approximation [103]. The coefficients α and β are regularization parameters to smooth and balance the reconstruction error. $\mathbf{L}_f$ and $\mathbf{L}_g$ are the normalized graph Laplacian matrices, with the definition of: $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}$. Tr represents the trace of a matrix. By optimizing the loss function, we can have the clustering results for the user-topic bipartite graph. To cluster the user-tweet bipartite graph, we utilize the cluster for tweets based on the clustering results of users. If one tweet belongs to different clusters, we use the majority vote strategy to choose a proper placement, which will be the

clustering result for user and tweet bipartite graph $\mathbf{B}_u$. The detailed optimization algorithm to iteratively update the matrices in Eq. (6.2) will be explained in Section 6.5.

6.4.3 Sentiment Analysis

As our goal is to extract open source intelligence from each community, we aggregate the tweets based on their cluster labels. Then, we run a sentiment analysis with a mini-batch algorithm when running the full-batch algorithms is intractable. We use the sentiment analysis library such as Textblob [104] to provide a quantitative result for the polarity in one cluster. Textblob is one of the commonly used libraries for processing textual data [104]. It provides APIs to handle natural language processing tasks including text cleaning and sentiment analysis. To get the polarity of a cluster, we measure the percentage of positive, neutral and negative tweets in that cluster. This way we can figure out the overall attitudes of the users in one cluster for the COVID-19 related events.

6.5 NMFR Updating Algorithm

In this section, we focus on solving the objective function that is formulated as a minimization problem in Eq. (6.2), where $\mathbf{F}_h$ and $\mathbf{G}_h$ are the cluster indicator matrices. The multiplication result of $\mathbf{B}_h\mathbf{B}^\top{}_h$ is a symmetric matrix. From [103], we know that the loss function can be

transformed to the following problem:

$$\begin{aligned} &\min ||\mathbf{B}_h\mathbf{B}^\top{}_h - \mathbf{F}_h\mathbf{S}_h\mathbf{G}_h||_F^2 \\ &+\alpha||\mathbf{F}_h - \mathbf{X}_f\mathbf{Y}_f||_F^2 + \beta||\mathbf{G}_h^\top - \mathbf{X}_g\mathbf{Y}_g||_F^2 \\ &\text{s.t. } \mathbf{F_h} \in \Psi^{n\times d}, \mathbf{G_h} \in \Psi^{k\times n}, \mathbf{Y}_g^\top\mathbf{Y}_g = \mathbf{I}, \mathbf{Y}_f^\top\mathbf{Y}_f = \mathbf{I} \end{aligned} \tag{6.5}$$

where $\mathbf{X}_g = \mathbf{W}_f^{-1/2}\mathbf{D}_g^{-1/2}$ and $\mathbf{X}_f = \mathbf{D}_f^{-1/2}\mathbf{W}_f^{-1/2}$, $\mathbf{Y}_f$ and $\mathbf{Y}_g$ are arbitrary orthonormal matrices.

To get $\mathbf{S}_h$, we can fix $\mathbf{F}_h$ and $\mathbf{G}_h$. Then, only the first term in Eq. (6.5) can affect the minimization process and the rest two terms are constants. Accordingly, we can calculate $\mathbf{S}_h$ by setting the derivative of the Eq. (6.5) to zero, and obtain Eq. (6.6):

$$\mathbf{S}_h = (\mathbf{F}_h^\top\mathbf{F}_h)^{-1}\mathbf{F}_h^\top\mathbf{B}_h\mathbf{B}^\top{}_h\mathbf{G}_h^\top(\mathbf{G}\mathbf{G}_h^\top)^{-1} \tag{6.6}$$

Next, if we fix $\mathbf{F}_h$, $\mathbf{G}_h$ and $\mathbf{S}_h$, we can obtain $\mathbf{Y}_f$ and $\mathbf{Y}_g$ by doing the singular value decomposition to $\mathbf{X}_f^\top\mathbf{F}_h$ and $\mathbf{X}_g^\top\mathbf{G}_h^\top$.

Similarly, we can fix $\mathbf{F}_h$, $\mathbf{S}_h$ and $\mathbf{Q}_g$ and find that the second term in Eq. (6.5) is constant. As a result, we only need to optimize the first and the third terms. Since $\mathbf{G}_h$ is the cluster indicator matrix, it can be obtained in the following equation:

$$\mathrm{Tr}(\mathbf{M}^\top\mathbf{Z} - \mathbf{M}^\top\mathbf{M}\mathbf{G} - \beta\mathbf{G} + \beta\mathbf{Y}^\top) = 0 \tag{6.7}$$

where:

$$\mathbf{M} = \mathbf{F}_h\mathbf{S}_h, \mathbf{Z} = \mathbf{B}_h\mathbf{B}^\top{}_h, \mathbf{G} = \mathbf{G}_h\mathbf{Y} = \mathbf{X}_g\mathbf{Y}_g.$$

To calculate $\mathbf{F}_h$, we can fix $\mathbf{G}_h$, $\mathbf{S}_h$ and $\mathbf{Q}_f$. Then, the third term in Eq. (6.5) is constant and we only need to minimize the first and the second term. Now, the cluster indicator matrix F_h can be obtained in the following equation:

$$\mathrm{Tr}(\mathbf{Z}\mathbf{M}^\top - \mathbf{F}\mathbf{M}\mathbf{M}^\top - \alpha\mathbf{F} + \alpha\mathbf{Y}) = 0 \tag{6.8}$$

where:

$$\mathbf{M} = \mathbf{S}_h\mathbf{G}_h, \mathbf{Z} = \mathbf{B}_h{\mathbf{B}^\top}_h, \mathbf{F} = \mathbf{F}_h\mathbf{Y} = \mathbf{X}_f\mathbf{Y}_f.$$

Algorithm 2 NMFR Updating (MNFRU) Algorithm

Input : data matrix $\mathbf{B}_h$, parameters α, β
Output : $\mathbf{F}_h, \mathbf{S}_h, \mathbf{G}_h$

1: initialize $\mathbf{F}_h, \mathbf{G}_h$ with random class indicator matrices;
2: calculate $\mathbf{X}_f$and$\mathbf{X}_g$
3: **while** it does not converge:
    update $\mathbf{S}_h$ according to Eq. (6.6);
    calculate $\mathbf{Y}_f$ by doing singular value decomposition to $\mathbf{X}_f^\top\mathbf{F}_h$;
    calculate $\mathbf{Y}_g$ by doing singular value decomposition $\mathbf{X}_g^\top\mathbf{G}_h^\top$;
    update $\mathbf{G}_h$ according to Eq. (6.7);
    update $\mathbf{F}_h$ according to Eq. (6.8);
4: **return** $\mathbf{F}_h$, $\mathbf{S}_h$ and $\mathbf{G}_h$

Algorithm 2 shows the pseudocode for the proposed NMFR Updating (NMFRU) algorithm, which can cluster the graph in the phase two of the TGC-PDA framework. The basic idea of the proposed NMFRU is to fix some factors and update one parameter at a time. Here, we start with one parameter that appears least frequently in the equation and iteratively update the matrices.

6.6 Experimental Results

In this section, we analyze our experimental results and the performance of TGC-PDA. We also compare NMFRU with the well-known clustering methods such as Kmeans, NMF and the commonly used variants, including Semi-NMF (SNMF) [105], Orthogonal NMTF (ONMTF) [106].

6.6.1 Dataset

We evaluate the performance of TGC-PDA with real Twitter dataset about "Covid-19" collected between Feb 15th, 2020 and September 30th 2020. To get the tweet data, we wrote a python program to crawl the tweets and the users who liked them. Multiple hashtag keywords such as #COVID19, #coronavirus, #covid, covid pandemic and #COVID20 are used to ensure we can get a large dataset. Since the free Twitter API we use has rate limits and it restricts the number of retrieved tweets during each login access, we have to crawl the data for several months. After removing the duplicate and non-English posts, we obtain 18,327 tweets, with 752,649 users who interacted with the tweets. Some users only interacted with one tweet in our dataset, which are identified as "less interactive" users and excluded. After the data cleanup, we have 301,982 users left.

6.6.2 Experimental Setup

As all the clustering methods (i.e., Kmeans, NMF, SNMF, ONMTF and our NMFRU) have one or more parameters to be tuned, to make the comparison fair, we run these methods under different parameters and choose the best result for each algorithm. In addition, we

set the number of clusters as the true number of classes for all clustering algorithms on the dataset. In specific, for Kmeans and NMF algorithms, the hyperparameter is k (number of clusters). If k is given, no other parameters would be needed. In NMFR, we have two hyperparameters: α and β. To find a proper value for these parameters, we plot a loss-value curve, with value ranging from 0.1 to 1000. Then, the α, β values can be found by scanning the plot. Since our data size is relative large and cannot be completely labelled manually, we randomly choose 5% of the data to label and use the result tested by sample data as the framework result.

6.6.3 Evaluation Metrics

To evaluate the clustering result, we use the widely used standard metrics including the clustering accuracy, cluster purity and normalized mutual information.

For the clustering accuracy, we compare the outputted clusters $c \in C_{out}$ with the ground truth labelled data $g \in C_{ground}$. The accuracy is defined as follows:

$$\text{Accuracy}(C_{out}, C_{ground}) = \frac{1}{n} \sum \delta(g_i, map(c_i)) \tag{6.9}$$

where n is total number of data samples, $\delta(a, b)$ is the delta function, in which the value equals one if $a = b$ and equals zero otherwise. The $map(c_i)$ is a mapping function that maps each cluster label c_i to the same label in the ground truth data. We can use Kuhn-Munkres algorithms to find the best mappings [107].

For the cluster purity, we compare the cluster output $c \in C_{out}$ with the ground truth labelled data $g \in C_{ground}$. The purity of the cluster result is calculated by the following

formula:

$$\text{Purity}(C_{out}, C_{ground}) = \frac{1}{n} \sum_{c \in C} \max_{g \in C_{ground}} (c \cap g) \tag{6.10}$$

where n is the number of data samples. A perfect clustering result has a purity of one, and a bad one has the purity value close to zero.

For the normalized mutual information (NMI), we compare the cluster output $c \in C_{out}$ with the ground truth labelled data $g \in C_{ground}$. The NMI is defined as:

$$\text{NMI}(C_{out}, C_{ground}) = \frac{2 \times I(C_{out}, C_{ground})}{[H(C_{out}) + H(C_{ground})]} \tag{6.11}$$

where H$(\cdot)$ represents the entropy, and $I(C_{out}, C_{ground})$ denotes the mutual information between C_{out} and C_{ground}. A higher NMI value means the better clustering result. To obtain a less biased estimation of the framework, we run NMFRU algorithms twenty times and take the average result for each model.

6.6.4 Results and Discussion

Table 6.2 shows the comparison between NMFRU and several baseline models such as Kmeans, NMF, SNMF and ONMTF. When applying these baseline models to our data, we do not embed the topic nodes to user nodes. Instead, we use the user and tweet bipartite graph to calculate the clustering result. The matrix form of the bipartite graph is the columns and rows correspond to the two set of vertices, with each entry corresponding to an edge between a column and a row. From Table 6.2, we can see that NMFRU achieves the best performance in terms of accuracy, purity and NMI. This is because our bipartite graph

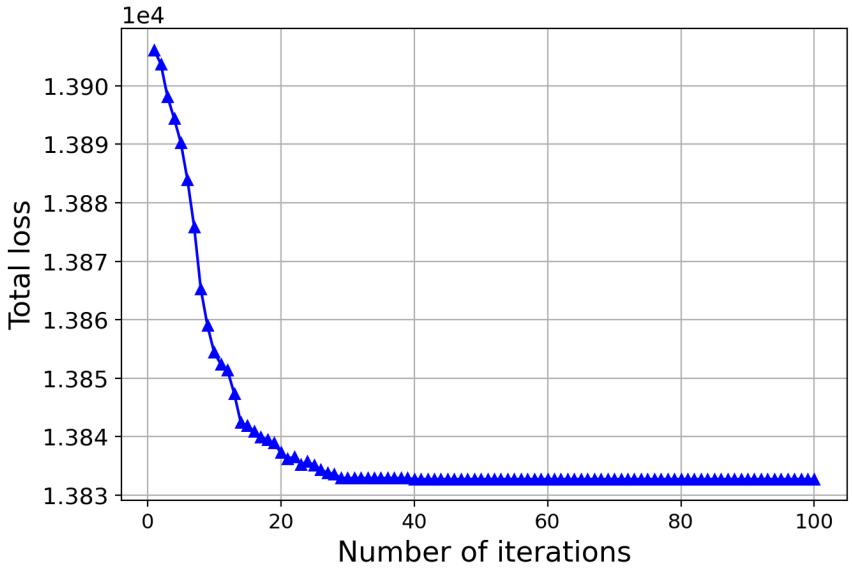

Figure 6.7: Total loss with different number of iterations.

Table 6.2: Performance results of classifiers

Methods	**Accuracy**	**Purity**	**NMI**
Kmeans	0.613	0.549	0.513
NMF	0.583	0.536	0.493
SNMF	0.627	0.562	0.534
ONMTF	0.674	0.578	0.557
NMFRU	0.706	0.617	0.621

is created based on our tripartite graph model, and it embedded more information than the plain bipartite graph. We also utilize the tri-factorization and locality preserved schemes, which can further improve the performance.

We study the average convergence time of our framework in Fig. 6.7. When the number of iterations is around 23, our framework tends to converge with a total loss of 6.2, which

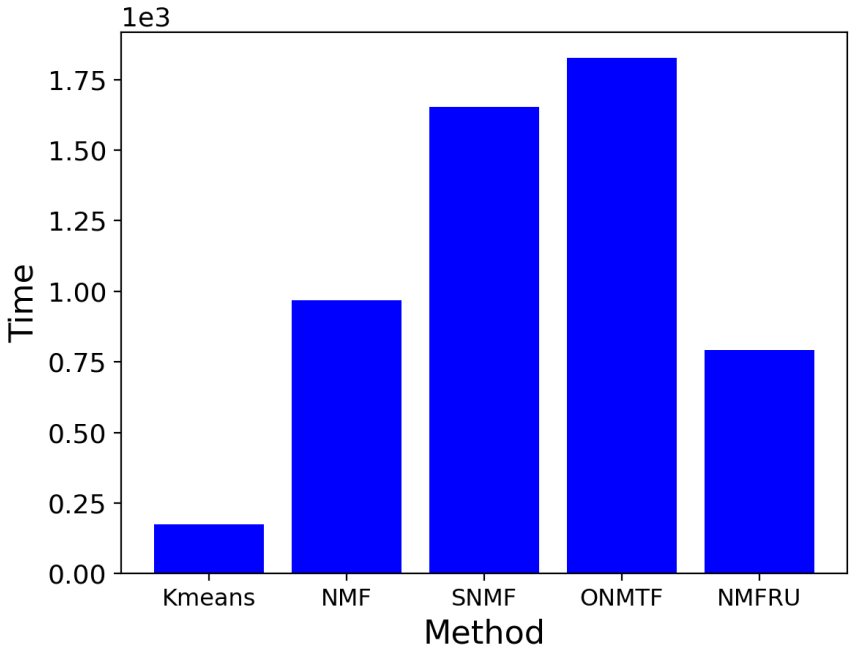

Figure 6.8: Convergence time (second) of methods.

shows that the calculation of NMFRU is fast. Meanwhile, when comparing the convergence time by the different baseline methods in Fig. 6.8, we can see that NMFRU is slower than Kmeans but faster than other baseline clustering methods. It is because we do fewer matrix multiplication operations in NMFRU, hence saving some running time. Therefore, TGC-PDA that utilizes NMFRU as the core clustering algorithm can be used for a large dataset.

As for the polarity of the communities, Table 6.3 shows the largest ten communities with its polarity ratio. From the Table 6.3, we find that the neutral ratio is quite high among all topics. In order to figure out the rationale here, we manually examined 1,000 posts and found that there are lots of media or government agencies (e.g. CNN and CDC) that use Twitter to publish the real-time news and the latest policy. These tweets tend to be retweeted many times. Obviously, such tweets are more likely to be considered neutral.

Table 6.3: Largest ten communities with its polarity ratio

Keywords	**Positive**	**Neutral**	**Negative**
marketcrash2021	18.2%	48.7%	33.1%
maskshortage	14.1%	41.6%	44.3%
death	4.1%	73.1%	22.8%
NYbreak	12.2%	57.5 %	30.3%
antibody	30.5%	41.3 %	28.2%
stimulus	32.6%	41.7%	25.7%
testing	32.7%	38.9%	28.4%
vaccine	20.4%	61.2%	18.4%
symptoms	26.3%	48.9%	24.8%
stayathome	23.6%	51.8%	24.6%

6.7 Chapter Summary

The outbreak of COVID-19 makes the whole world chaotic. People often search for real-time news and ventilate their emotions through the Internet. Online social networks (OSNs) are widely used for opinions sharing, news publishing and information spreading. The large useful data from OSNs can be leveraged to help public officials and government make better decisions. In this paper, we build a framework of Tripartite Graph Clustering for Pandemic Data Analysis (TGC-PDA) to utilize Twitter data to monitor and automatically collect the voice of the people during COVID-19 pandemic. The TGC-PDA framework takes advan-

tage of the characteristics of the Twitter users and tweets network structure to effectively analyze the community structures and sentiments. It enables us to extract the open source intelligence from each community, which could be utilized to track people's feedbacks and opinions towards the Coronavirus pandemic events. Our work currently is a pioneering work and it only focused on English-language tweets. It would be feasible to extend our work to handle tweets in other languages. Similar techniques can be applied to other online and publicly available social media platforms such as Reddit. Since a tweet may contain not only text, but also embedded hyperlinks, images or even videos, it would be interesting and challenging to explore more information from them. Moreover, some events during COVID-19 are time-sensitive, it would be also interesting to study the tweets from the perspective of time-series analysis.

CHAPTER 7
DISCUSSION

In this dissertation, we study graph mining and its applications in online social networks. Similarity measurement lays a solid foundation for graph mining problems, which provide a point of view to solve interesting and challenging problems such as community detection, query and ranking and link prediction. With the developed method for graph mining problems, we can apply them to real-world occurrences. That is the reason we applied the similarity measurement method to Online Social Networks (OSNs) to rank the top-k nodes and predict the links.

In our current work, the single source algorithm is developed, the key idea is the power iteration method. When the graph expands, the computation would become super slow. To optimize the algorithm, one strategy is trading accuracy for efficiency. If we can randomize the calculation, for example, with Monte Carlo methods, the computation complexity could decrease. In [108, 109], the authors introduced the Monte Carlo Methods and how to apply them to the Markov chain. This shows us the feasibility of employing Monte Carlo methods to the CoSimRank algorithms. In our second work, we utilize the traditional machine learning method with manually selected features, which provides high accuracy with much background knowledge. A potential work is to use deep learning models to take the place of feature selection. Researchers such as the authors in [110, 111] provide many deep learning based tools, which shed light on combining both image and text for user credibility analysis. For our third work, we utilize the Twitter text exclusively. It would be feasible to try to

include other data such as embedded hyperlinks, so that we can extract the opinions of the link basing on its linked text. Then, we can take both tweets and link text into consideration.

With the scale of these networks explosively increases from millions to even billions of nodes, it becomes inefficient and ineffective to apply traditional machine learning models to the simple representation of the network. Moreover, the network is sparse, which makes it harder to generalize in statistical learning. The goal of network representation learning is to learn low-dimensional representations of network, while preserving network information, such as topology structure, node content or labels, edge attributes and other side information. For example, the original graph representation we proposed in Chapter 6 is a tripartite graph, which means we will need 3-d array or tensor to represent three types of nodes. With the method that we developed in Chapter 6.4.1, we decrease the input data dimension. This not only eliminates the requirement for developing intricate algorithms in the original network, but also makes large network analysis feasible. Thus, it will be meaningful to research on a method to represent and preserve the network structure so that it can be exploited by machine learning models.

Since our current work focuses on graph mining in the social networking domain, we can transform the OSN problems into graph problems. Similarly, a biological network can be represented as graphs and edges, which means our existing algorithms and tools from graph mining could be applied to biological networks easily. For instance, we can use our link prediction methods to predict interactions between genes and proteins in a biological network. With a better understanding of the complex biological activities between proteins

or genes, infection, disease progression and many other biomedical problems could be further understood.

CHAPTER 8
CONCLUSION

In this dissertation, we study the methods in obtaining open-source intelligence in Online Social Networks. As online social networks can be naturally represented by graphs, the problems in social networks can be transformed into graph problems. Analyzing Online Social Networks is a process of investigating social structure through the use of networks and graph theory. For a social network, the nodes can represent individuals, accounts or entities. The links or edges that connect the nodes represent the interaction or relationship between nodes. Thus, the representations of the social networks could be diverse. They can be homogeneous, heterogeneous, unipartite, or multipartite. In the meantime, the graph can be undirected (edges point both ways) or directed (edges point from one node to another, but not vice versa). Besides, edges can have different weights or not, which differentiates weighted graphs from unweighted graphs. Different representations of a social network lead to various methods to discover knowledge from it.

Graph mining has become increasingly critical with broad applications, which have demand on analyzing large structured data. Thus, graph mining has become a crucial subdomain in data mining. The graph mining methods target mining useful knowledge from structure information of the data with better performance and innovation.

Firstly, this dissertation introduces the characteristics of online social networks and their real-world universality and importance. The network structure and representation are discussed and explained. Several primitive problems in graph mining as demonstrated as well.

They provide the overall introduction and mathematical foundation of the dissertation.

Secondly, the dissertation proposes a corresponding algorithm for proximity measures in online social networks. The proximity measure utilizes second-order knowledge of the graphs, and can provide higher prediction accuracy for applications such as ranking and link prediction.

Thirdly, the dissertation designs a framework for content and influencer credibility in online social networks. The texts and images are jointly considered and properly balanced. The proposed framework works as an approach to deal with the challenges for trustworthiness in the current online social networking environment in the application level.

Fourthly, the dissertation investigates extracting open source intelligence from online social networks to help with Coronavirus pandemic-related decision making for governors and policymakers. As Twitter provides channels for ordinary people discussing about policies or events in real-time, the large discussion conversation can be utilized in our work. We propose a framework to automatically gather and analyze COVID-19 related events.

All the proposed solutions are thoroughly discussed and validated through extensive evaluations. Generally, our dissertation provides a body of solutions for gathering open-source intelligence for social networks with graph mining methods. These solutions could comprehensively cover the challenges for information loss caused by the simplified graph representation. We believe the study in this dissertation will work as a reference for the tasks of graph mining in online social networks. Finally, this dissertation will also be an inspiration to stimulate the subsequent efforts towards the publication of ground truth online

social network data.

REFERENCES

[1] *Wikipedia:COVID-19 pandemic*, 2020. [Online]. Available: https://en.wikipedia.org/wiki/COVID-19_pandemic

[2] "Cambridge dictionary," 2021. [Online]. Available: https://dictionary.cambridge.org/us/dictionary/english/network

[3] L. Garton, C. Haythornthwaite, and B. Wellman, "Studying Online Social Networks," *Journal of Computer-Mediated Communication*, vol. 3, no. 1, 06 1997, jCMC313. [Online]. Available: https://doi.org/10.1111/j.1083-6101.1997.tb00062.x

[4] *statista: Global No.1 Business Data Platform*, 2020. [Online]. Available: https://www.statista.com/

[5] *The New York Times: Social Media's Globe-Shaking Power*, 2016. [Online]. Available: https://www.nytimes.com/2016/11/17/technology/social-medias-globe-shaking-power.html?_r=0

[6] J. Han and M. Kamber, "Data mining concepts and techniques second edition," *The Morgan Kaufmann Series in Data Management Systems*, 2006.

[7] S. Fortunato and D. Hric, "Community detection in networks: A user guide," *Physics reports*, vol. 659, pp. 1–44, 2016.

[8] M. E. Newman, "Community detection and graph partitioning," *EPL (Europhysics Letters)*, vol. 103, no. 2, p. 28003, 2013.

[9] D. Zhou, E. Manavoglu, J. Li, C. L. Giles, and H. Zha, "Probabilistic models for

discovering e-communities," in *Proceedings of the 15th international conference on World Wide Web*, 2006, pp. 173–182.

[10] M. Newman, "Spectral methods for community detection and graph partitioning," *Physical Review E*, vol. 88, no. 4, p. 042822, 2013.

[11] M. E. Newman, "Modularity and community structure in networks," *Proceedings of the national academy of sciences*, vol. 103, no. 23, pp. 8577–8582, 2006.

[12] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web," 1999.

[13] H. Tong, C. Faloutsos, and J.-Y. Pan, "Fast random walk with restart and its applications," *the 6th International Conference on Data Mining*, pp. 613–622, 2006.

[14] G. Jeh and J. Widom, "Simrank: a measure of structural-context similarity," *Proceedings of the 8th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 538–543, 2002.

[15] X. Yang, Z. Zhang, and K. Wang, "Scalable collaborative filtering using incremental update and local link prediction," in *Proceedings of the 21st ACM international conference on Information and knowledge management*, 2012, pp. 2371–2374.

[16] A. Papadimitriou, P. Symeonidis, and Y. Manolopoulos, "Fast and accurate link prediction in social networking systems," *Journal of Systems and Software*, vol. 85, no. 9, pp. 2119–2132, 2012.

[17] R. Real and J. M. Vargas, "The probabilistic basis of jaccard's index of similarity," *Systematic biology*, vol. 45, no. 3, pp. 380–385, 1996.

[18] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *Journal of the American society for information science and technology*, vol. 58, no. 7, pp. 1019–1031, 2007.

[19] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953.

[20] X. Liao, Y. Wu, and X. Cao, "Second-order cosimrank for similarity measures in social networks," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–6.

[21] S. Wasserman and K. Faust, *Social network analysis: Methods and applications*. Cambridge University Press, 1994.

[22] C. Kong, G. Luo, L. Tian, and X. Cao, "Disseminating authorized content via data analysis in opportunistic social networks," *Big Data Mining and Analytics*, vol. 2, pp. 12–24, 2019.

[23] M. R. Bouadjenek, H. Hacid, and M. Bouzeghoub, "Sopra: A new social personalized ranking function for improving web search," *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pp. 861–864, 2013.

[24] P. Bedi and C. Sharma, "Community detection in social networks," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 6, no. 3, pp. 115–135, 2016.

[25] Y. Wu, R. Jin, J. Li, and X. Zhang, "Robust local community detection: On free rider

effect and its elimination," *Proceedings of the VLDB Endowment*, vol. 8, no. 7, pp. 798–809, 2015.

[26] L. Lü and T. Zhou, "Link prediction in complex networks: A survey," *Physica A*, vol. 390, no. 6, pp. 1150–1170, 2011.

[27] L. Li, C. Li, H. Chen, and X. Du, "Mapreduce-based simrank computation and its application in social recommender system," *2013 IEEE International Congress on Big Data*, pp. 133–140, 2013.

[28] P. Li, H. Liu, J. X. Yu, J. He, and X. Du, "Fast single-pair simrank computation," *Proceedings of the 2010 SIAM International Conference on Data Mining*, pp. 571–582, 2010.

[29] Y. Shao, B. Cui, L. Chen, M. Liu, and X. Xie, "An efficient similarity search framework for simrank over large dynamic graphs," *Proceedings of the VLDB Endowment*, vol. 8, no. 8, pp. 838–849, 2015.

[30] S. Rothe and H. Schütze, "Cosimrank: A flexible & efficient graph-theoretic similarity measure." *ACL (1)*, pp. 1392–1402, 2014.

[31] W. Yu and J. A. McCann, "Co-simmate: Quick retrieving all pairwise co-simrank scores," *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, vol. 2, pp. 327–333, 2015.

[32] Y. Wu, Y. Bian, and X. Zhang, "Remember where you came from: on the second-order random walk based proximity measures," *Proceedings of the VLDB Endowment*,

vol. 10, no. 1, pp. 13–24, 2016.

[33] Y. Wu, X. Zhang, Y. Bian, Z. Cai, X. Lian, X. Liao, and F. Zhao, "Second-order random walk-based proximity measures in graph analysis: formulations and algorithms," *The VLDB Journal*, vol. 27, no. 1, pp. 127–152, 2018.

[34] W. Yu and F. Wang, "Fast exact cosimrank search on evolving and static graphs," *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pp. 599–608, 2018.

[35] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *Society for Industrial and Applied Mathematics review*, vol. 51, no. 3, pp. 455–500, 2009.

[36] A. N. Langville and C. D. Meyer, *Google's PageRank and Beyond: The Science of Search Engine Rankings*. Princeton University Press, 2006.

[37] K. Madduri and D. A. Bader. (2006) Gtgraph: A suite of synthetic random graph generators. [Online]. Available: http://www.cse.psu.edu/~kxm85/software/GTgraph/

[38] Statista, "Most popular social networks worldwide as of january 2019, ranked by number of active users," 2019. [Online]. Available: https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/

[39] T. Clarke, 2018. [Online]. Available: https://blog.hootsuite.com/black-hat-social-media/

[40] L. Fair, "Three ftc actions of interest to influencers," 2017. [Online]. Available: https://www.ftc.gov/news-events/blogs/business-blog/2017/09/three-ftc-actions-interest-influencers

[41] T. Martin, 2016. [Online]. Available: https://www.instagram.com/p/BCgvq7_QN8r/?hl=en

[42] M. Grbovic, N. Djuric, V. Radosavljevic, F. Silvestri, and N. Bhamidipati, "Context- and content-aware embeddings for query rewriting in sponsored search," in *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. ACM, 2015, pp. 383–392.

[43] J. Zhao, G. Qiu, Z. Guan, W. Zhao, and X. He, "Deep reinforcement learning for sponsored search real-time bidding," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 1021–1030.

[44] Z. Lu, H. Zhou, V. O. Li, and Y. Long, "Pricing game of celebrities in sponsored viral marketing in online social networks with a greedy advertising platform," pp. 1–6, 2016.

[45] C. J. Plume and E. L. Slade, "Sharing of sponsored advertisements on social media: A uses and gratifications perspective," *Information Systems Frontiers*, vol. 20, no. 3, pp. 471–483, 2018.

[46] Z. Lu, V. O. Li, and Q. Shuai, "Price competition of spreaders in profit-maximizing sponsored viral marketing," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 4, pp. 931–941, 2018.

[47] M. Scholz, C. Brenner, and O. Hinz, "Akegis: automatic keyword generation for sponsored search advertising in online retailing," *Decision Support Systems*, 2019.

[48] D. Mutum and Q. Wang, "Consumer generated advertising in blogs," in *Handbook of*

research on digital media and advertising: User generated content consumption. IGI Global, 2011, pp. 248–261.

[49] B. Chacon, "How to properly disclose sponsored instagram posts according to the ftc," 2017. [Online]. Available: https://later.com/blog/sponsored-instagram-posts/

[50] B. Sriram, D. Fuhry, E. Demir, H. Ferhatosmanoglu, and M. Demirbas, "Short text classification in twitter to improve information filtering," in *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval.* ACM, 2010, pp. 841–842.

[51] M. Skowron, M. Tkalčič, B. Ferwerda, and M. Schedl, "Fusing social media cues: personality prediction from twitter and instagram," in *Proceedings of the 25th international conference companion on world wide web.* International World Wide Web Conferences Steering Committee, 2016, pp. 107–108.

[52] pc.net, 2019. [Online]. Available: https://pc.net/emoticons/

[53] I. Pennebaker Conglomerates, "Linguistic inquiry and word count," 2015. [Online]. Available: https://liwc.wpengine.com/

[54] M. Liberman, "Alphabetical list of part-of-speech tags used in the penn treebank project," 2019. [Online]. Available: https://www.ling.upenn.edu/courses/Fall-2003/ling001/penn-treebank-pos.html

[55] B. Liu and L. Zhang, "A survey of opinion mining and sentiment analysis," in *Mining text data.* Springer, 2012, pp. 415–463.

[56] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in

international Conference on computer vision & Pattern Recognition (CVPR'05), vol. 1. IEEE Computer Society, 2005, pp. 886–893.

[57] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[58] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[59] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[60] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[61] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[62] K. P. Bennett and E. J. Bredensteiner, "Duality and geometry in svm classifiers," in *ICML*, vol. 2000, 2000, pp. 57–64.

[63] H. W. Kuhn and A. W. Tucker, "Nonlinear programming, in (j. neyman, ed.) proceedings of the second berkeley symposium on mathematical statistics and probability," 1951.

[64] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[65] *Everyone Included: Social Impact of COVID-19*, 2020. [Online]. Available:

https://www.un.org/development/desa/dspd/everyone-included-covid-19.html

[66] *Domestic Travel During the COVID-19 Pandemic*, 2020. [Online]. Available: https://www.cdc.gov/coronavirus/2019-ncov/travelers/travel-during-covid19.html

[67] *Travelers Prohibited from Entry to the United States*, 2020. [Online]. Available: https://www.cdc.gov/coronavirus/2019-ncov/travelers/from-other-countries.html

[68] K. Cohen, *Tokyo 2020 Olympics officially postponed until 2021*, 2020. [Online]. Available: https://www.espn.com/olympics/story/_/id/28946033/tokyo-olympics-officially-postponed-2021

[69] *Wikipedia:RNA virus*, 2021. [Online]. Available: https://en.wikipedia.org/wiki/RNA_virus

[70] *How does fake news of 5G and COVID-19 spread worldwide?*, 2021. [Online]. Available: https://www.medicalnewstoday.com/articles/5g-doesnt-cause-covid-19-but-the-rumor-it-does-spread-like-a-virus

[71] L. Chang, W. Li, L. Qin, W. Zhang, and S. Yang, "pscan: Fast and exact structural graph clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 2, pp. 387–401, 2017.

[72] R. EL BACHA and T. T. Zin, "Ranking of influential users based on user-tweet bipartite graph," in *2018 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*. IEEE, 2018, pp. 97–101.

[73] A. Rodríguez, C. Argueta, and Y.-L. Chen, "Automatic detection of hate speech on facebook using sentiment and emotion analysis," in *2019 International Conference on*

Artificial Intelligence in Information and Communication (ICAIIC). IEEE, 2019, pp. 169–174.

[74] J. Zhou and C. Kwan, "Missing link prediction in social networks," in *International Symposium on Neural Networks*. Springer, 2018, pp. 346–354.

[75] A. Reyes-Menendez, J. R. Saura, and C. Alvarez-Alonso, "Understanding# worldenvironmentday user opinions in twitter: A topic-based sentiment analysis approach," *International journal of environmental research and public health*, vol. 15, no. 11, p. 2537, 2018.

[76] C. Tan, L. Lee, J. Tang, L. Jiang, M. Zhou, and P. Li, "User-level sentiment analysis incorporating social networks," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2011, pp. 1397–1405.

[77] A. Giachanou and F. Crestani, "Like it or not: A survey of twitter sentiment analysis methods," *ACM Computing Surveys (CSUR)*, vol. 49, no. 2, pp. 1–41, 2016.

[78] R. R. Iyer, J. Chen, H. Sun, and K. Xu, "A heterogeneous graphical model to understand user-level sentiments in social media," *arXiv preprint arXiv:1912.07911*, 2019.

[79] H. Deng, J. Han, H. Li, H. Ji, H. Wang, and Y. Lu, "Exploring and inferring user–user pseudo-friendship for sentiment analysis with heterogeneous networks," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 7, no. 4, pp. 308–321, 2014.

[80] C. A. Phillips, "Multipartite graph algorithms for the analysis of heterogeneous data," 2015.

[81] D. Zhou, S. Zhang, M. Y. Yildirim, S. Alcorn, H. Tong, H. Davulcu, and J. He, "A local algorithm for structure-preserving graph cut," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 655–664.

[82] P. M. Comar, P.-N. Tan, and A. K. Jain, "A framework for joint community detection across multiple related networks," *Neurocomputing*, vol. 76, no. 1, pp. 93–104, 2012.

[83] Y. Sun, Y. Yu, and J. Han, "Ranking-based clustering of heterogeneous information networks with star network schema," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 797–806.

[84] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in neural information processing systems*, 2001, pp. 556–562.

[85] N. Gillis, "The why and how of nonnegative matrix factorization," *Regularization, Optimization, Kernels, and Support Vector Machines*, pp. 257–291, 2014.

[86] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[87] M. E. Wall, A. Rechtsteiner, and L. M. Rocha, "Singular value decomposition and principal component analysis," in *A practical approach to microarray data analysis*. Springer, 2003, pp. 91–109.

[88] C. Ding, T. Li, W. Peng, and H. Park, "Orthogonal nonnegative matrix t-factorizations for clustering," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006, pp. 126–135.

[89] D. Kim, S. Sra, and I. S. Dhillon, "Fast newton-type methods for the least squares nonnegative matrix approximation problem," in *Proceedings of the 2007 SIAM international conference on data mining*. SIAM, 2007, pp. 343–354.

[90] C.-J. Lin, "On the convergence of multiplicative update algorithms for nonnegative matrix factorization," *IEEE Transactions on Neural Networks*, vol. 18, no. 6, pp. 1589–1596, 2007.

[91] J. Kim and H. Park, "Toward faster nonnegative matrix factorization: A new algorithm and comparisons," in *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 2008, pp. 353–362.

[92] F. Wang and P. Li, "Efficient nonnegative matrix factorization with random projections," in *Proceedings of the 2010 SIAM International Conference on Data Mining*. SIAM, 2010, pp. 281–292.

[93] M. Annett and G. Kondrak, "A comparison of sentiment analysis techniques: Polarizing movie blogs," in *Conference of the Canadian Society for Computational Studies of Intelligence*. Springer, 2008, pp. 25–35.

[94] R. Hillmann and M. Trier, "Sentiment polarization and balance among users in online social networks," 2012.

[95] M. Del Vicario, G. Vivaldo, A. Bessi, F. Zollo, A. Scala, G. Caldarelli, and W. Quattrociocchi, "Echo chambers: Emotional contagion and group polarization on facebook," *Scientific reports*, vol. 6, p. 37825, 2016.

[96] S. M. Mohammad, X. Zhu, S. Kiritchenko, and J. Martin, "Sentiment, emotion, pur-

pose, and style in electoral tweets," *Information Processing & Management*, vol. 51, no. 4, pp. 480–499, 2015.

[97] K. Chakraborty, S. Bhattacharyya, R. Bag, and A. Hassanien, "Sentiment analysis on a set of movie reviews using deep learning techniques," in *Social Network Analytics—Computational Research Methods and Techniques*. Elsevier, 2018.

[98] K. Sailunaz and R. Alhajj, "Emotion and sentiment analysis from twitter text," *Journal of Computational Science*, vol. 36, p. 101003, 2019.

[99] H. Meisheri, K. Ranjan, and L. Dey, "Sentiment extraction from consumer-generated noisy short texts," in *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 2017, pp. 399–406.

[100] A. S. M. Alharbi and E. de Doncker, "Twitter sentiment analysis with a deep neural network: An enhanced approach using user behavioral information," *Cognitive Systems Research*, vol. 54, pp. 50–61, 2019.

[101] M. Wang, C. Wang, J. X. Yu, and J. Zhang, "Community detection in social networks: an in-depth benchmarking study with a procedure-oriented framework," *Proceedings of the VLDB Endowment*, vol. 8, no. 10, pp. 998–1009, 2015.

[102] D. Cai, X. He, X. Wu, and J. Han, "Non-negative matrix factorization on manifold," in *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 2008, pp. 63–72.

[103] H. Wang, F. Nie, H. Huang, and F. Makedon, "Fast nonnegative matrix tri-factorization for large-scale data co-clustering," in *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[104] *TextBlob: Simplified Text Processing*, 2020. [Online]. Available: https://textblob.readthedocs.io/en/dev/

[105] C. H. Ding, T. Li, and M. I. Jordan, "Convex and semi-nonnegative matrix factorizations," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 1, pp. 45–55, 2010.

[106] H. Abe and H. Yadohisa, "Orthogonal nonnegative matrix tri-factorization based on tweedie distributions," *Advances in Data Analysis and Classification*, vol. 13, no. 4, pp. 825–853, 2019.

[107] P. K. Shivaswamy and T. Jebara, "Permutation invariant svms," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 817–824.

[108] D. P. Kroese, T. Taimre, and Z. I. Botev, *Handbook of monte carlo methods*. John Wiley & Sons, 2013, vol. 706.

[109] D. J. C. Mackay, "Introduction to monte carlo methods," in *Learning in graphical models*. Springer, 1998, pp. 175–204.

[110] R. Socher, Y. Bengio, and C. D. Manning, "Deep learning for nlp (without magic)," in *Tutorial Abstracts of ACL 2012*, 2012, pp. 5–5.

[111] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.