3-26-2014

Mining the Online Social Network Data: Influence, Summarization, and Organization

Jingxuan Li
jli003@cs.fiu.edu

DOI: 10.25148/etd.FI14040816

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

MINING THE ONLINE SOCIAL NETWORK DATA: INFLUENCE, SUMMARIZATION, AND ORGANIZATION

A dissertation submitted in partial fulfillment of the

requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

by

Jingxuan Li

2014

To: Dean Amir Mirmiran
College of Engineering and Computing

This dissertation, written by Jingxuan Li, and entitled Mining the Online Social Network Data: Influence, Summarization, and Organization, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

Sundaraja Sitharama Iyengar

Shu-Ching Chen

Bogdan Carbunar

Debra VanderMeer

Tao Li, Major Professor

Date of Defense: March 26, 2014

The dissertation of Jingxuan Li is approved.

Dean Amir Mirmiran
College of Engineering and Computing

Dean Lakshmi N. Reddi
University Graduate School

Florida International University, 2014

DEDICATION

To my wife, my parents, and my daughter.

ACKNOWLEDGMENTS

I would like to thank my Ph.D. advisor, Dr. Tao Li. He offered me invaluable advice and financial support throughout my Ph.D. study in School of Computing and Information Sciences at Florida International University. Dr. Li taught me how to conduct research, gave me freedom to identify interesting research problems, and helped me find out solutions to those problems. I cannot thank Dr. Li enough.

I would also like to thank Dr. Sundaraja Sitharama Iyengar, Dr. Shu-Ching Chen, Dr. Bogdan Carbunar, and Dr. Debra VanderMeer for taking their time to serve on the committee and provide feedback for my dissertation work.

I extend my thanks to Dr. Tong Sun and Dr. Wei Peng in Xerox Research Center, who gave me help and support during my internship.

Special thanks to all of my co-authors. In addition, my thanks go to my collaborators including Dr. Ning Xie, Dr. Changjun Wu, and many others, who have provided me useful suggestions.

ABSTRACT OF THE DISSERTATION

MINING THE ONLINE SOCIAL NETWORK DATA: INFLUENCE, SUMMARIZATION, AND ORGANIZATION

by

Jingxuan Li

Florida International University, 2014

Miami, Florida

Professor Tao Li, Major Professor

Online Social Network (OSN) services provided by Internet companies bring people together to chat, share the information, and enjoy the information. Meanwhile, huge amounts of data are generated by those services (they can be regarded as the social media) every day, every hour, even every minute, and every second. Currently, researchers are interested in analyzing the OSN data, extracting interesting patterns from it, and applying those patterns to real-world applications. However, due to the large-scale property of the OSN data, it is difficult to effectively analyze it.

This dissertation focuses on applying data mining and information retrieval techniques to mine two key components in the social media data — users and user-generated contents. Specifically, it aims at addressing three problems related to the social media users and contents: (1) how does one organize the users and the contents? (2) how does one summarize the textual contents so that users do not have to go over every post to capture the general idea? (3) how does one identify the influential users in the social media to benefit other applications, e.g., Marketing Campaign?

The contribution of this dissertation is briefly summarized as follows. (1) It provides a comprehensive and versatile data mining framework to analyze the users and user-generated contents from the social media. (2) It designs a hierarchical co-clustering algorithm to organize the users and contents. (3) It proposes multi-document summarization methods to extract core information from the social

network contents. (4) It introduces three important dimensions of social influence, and a dynamic influence model for identifying influential users.

TABLE OF CONTENTS

CHAPTER PAGE

LIST OF TABLES

TABLE PAGE

LIST OF FIGURES

FIGURE PAGE

CHAPTER 1

INTRODUCTION AND MOTIVATION

The social network is a social structure of a set of individuals and the ties between them. Because of the social and informative property of the social network, the social network analysis, as a research topic, has attracted much attention from different domains, including economics, anthropology, biology, social psychology, physics, information science, etc. The analysis techniques of the social network are mainly coming from sociology, statistics, and mathematics [WF94], and they highly rely on the data in their own domain. It is worth noticing that, before the advent of the online social network (OSN) websites, collecting the social network data is believed to be difficult for most researchers due to the limitation of resources.

Recently, social network data collection is becoming easier because of the rapid development of the Online Social Network platform (e.g., Facebook, Twitter, and Google+). These platforms can be defined as social softwares, which help people interact/communicate with each other, or engage in the interaction. Considering the convenient accessibility of OSN through different digital appliances, e.g., smartphone, tablet, laptop, etc., OSN has become the news and updates sharing platform in addition to serving interactions. It is thus also named as social media. Due to its roles as both a communication channel and a media, social network data is everywhere. For example, there is data about friendship, affiliation, email, co-author, call, movie and music networks, etc. Meanwhile, the data collection is becoming trivial via the API and webpages provided by those OSN services.

Having addressed the research data set problem, the analysis over those data sets is turning out to be non-trivial, since on the one hand, they are usually large-scale and the traditional analytics methods cannot be applied on them directly; on the other hand, some analytics methods are borrowed from the other domains instead of social network, thus cannot accurately consider the properties of OSN.

In order to design prominent techniques to preprocess and analyze the large-scale OSN data, computer science is brought to the front, especially data mining and information retrieval. Notice that the strength of data mining is to build descriptive and predictive models for the data, while the advantage of information retrieval is to obtain the relevant information from a large collection of information resources. Thus, this research is going to employ data mining and information retrieval techniques to mine the OSN data.

There have already been different branches of research on the OSN data using data mining and information retrieval, including information diffusion and cascading [KKT03, KKT05, LMF+07], link prediction [LNK07, CMN08], experts and prominent actors identification [STLS06, DYB+07, CNN+10], search [AA05], trust and distrust on social network [GKRT04, DHP07], community detection [New06, GN02], etc.

1.1 Motivation to Conduct Research on Users and Contents

On the one hand, it can be observed from different branches of the research on the OSN data that all of these research works are surrounding the users and user-generated contents in OSN. For example, some researchers have studied the properties of the users in the online social network, and proposed methods to rank the influential users in OSN [CHBG10, RGAH10, WLJH10]. While some other work focuses on the study of information diffusion through the social network, the information can be in the form of blogs, posts, tweets, comments, tags, images, and URLs [GGLNT04, RMK11, YC10].

Notice that users are individuals who have registered on the social media services, and are generating and propagating the updates and news through the services. Meanwhile, users themselves have different types of social ties, such as parents, friends, classmates, comrades, couples, and so on. Thus, we conclude that users of the social media form OSN, and they are exchanging and propagating ev-

erything, which can be called social contents (posts, blogs, tweets, reviews, images, URLs, tags, etc.) through OSN. As a result, users and the contents flowing in the social network are two core components of OSN.

Researchers believe that through the studies of social media users and contents, (1) sociologists can have a better understanding of how people behave on OSN, evaluate and analyze the research results before the advent of OSN, and identify the difference between online social network and the social network in the real world; (2) economists can develop new economical models based on the OSN data; (3) biologists are aware of the connection between the epidemics and social network; (4) physicists can have insights about the phenomenons in the social network; (5) mathematicians and statisticians are going to develop or extend more mathematical models due to the motivation of generalizing the OSN analysis methods; (6) computer scientists and information scientists are going to design more interesting and practical algorithms in response to the analytical requirements of the OSN data.

On the other hand, at the same time when the academic research is on-going, the industry poses new requirements in studying the users and contents in the social media. (1) Marketing users seek tips and suggestions from the OSN data to increase the user engagement; (2) Big Internet companies are eager for tools to organize the information of users and contents in the social media; (3) Information providers cannot wait for providing their users with the appropriate personalized contents.

In a word, the research of social media users and contents is important for both academia and industry, and many researchers have taken the very first step. However, (1) most of the existing work over the social network users mining focuses on their own empirical purposes, and does not attempt to analyze users under different circumstances. In other words, it is hard to extend those works for users in different OSNs, for different purposes; (2) the existing works over the social

network contents are still immature, in terms of the inability to provide appropriate, accurate, concise, and meaningful summaries to serve different requirements of the audience; (3) few existing works explicitly organize the users and contents based on abundant relationships between different users, different contents, the user and the content; in addition, few of them attempt to improve the mining procedure of one type of data by incorporating the other type of data.

In order to address the above issues, this dissertation designs and provides a large-scale social network analytics framework, which aims at the further study of the user and content in OSN based on the existing research work. It bridges the gap between the user and content aspects of the OSN research via data mining and information retrieval techniques, and sheds light on deep understanding of the user and content, as well as the relationship between them. Moreover, this research improves the state of the art methods for mining the user and content in OSN by addressing the issues inherent to them.

Notice that this research will not cover every single existing research area mentioned before for the user and content in OSN. Rather, considering the large-scale property of the OSN data, the main aim of this research is two-fold: (1) identifying the most important and relevant information, in other words, finding out users and contents pertaining people's requirements, so that people do not have to go over every user or every piece of content in the OSN; (2) organizing both users and user-generated contents simultaneously into a data structure based on their intrinsic relationships, so that the future information retrieval and recommendation requirements can be fulfilled.

1.2 Research Problem Statement

As introduced before, the users and contents in the social media have attracted much research attention. However, due to the large-scale property of the data, people cannot go through every user and every piece of content to fulfill their information needs. Moreover, even though fully visiting of the users and contents is possible for some particular data sets, it is still highly time/effort consuming and not necessary, since usually a small key set of users and contents have the capability of representing the whole data set, and the efforts only need to be devoted to the small set. In other words, methods, which can tailor to the huge volume of social network data per different requirements and identify the most representative/significant information from the data are important.

This research will follow this direction to provide a comprehensive and versatile framework to mine and analyze the large-scale social network users and contents. In particular, this framework focuses on identifying "influential" users and summarizing contents in OSN by considering the existing methods as the basis, and proposing new models and algorithms to address a more general OSN "influential" users identification problem and a more meaningful OSN contents summarization problem. Furthermore, on top of dealing with the two problems related to the users and contents, respectively, this research is dedicated to utilize the relationship between the users and the contents in the social network to organize the OSN data, and explore the possibility of enriching the mining procedure of one particular type of data (users/contents) via the other type of data (contents/users).

To facilitate the understanding of capabilities of the proposed framework, several interesting questions, which are going to be answered by this framework in this research are listed below:

1. Given the specific requirement posed by people, who are the "influential" users in an OSN? Will they be influential in the future? Why?

2. How does one summarize the contents in OSN while not compromising with too much information loss or reducing the readability, so that users can quickly capture and begin to enjoy reading the general ideas about their interested topics?

3. What is the relationship between users and contents in the social network? How does one utilize these relationships to organize those involved users and contents?

1.3 Research Purpose and Significance

The ultimate goal of this dissertation is that with its assistance, everyone, no matter whether he/she is a researcher, a political leader, a CEO in the company, or an ordinary individual, he/she can mine the data, then identify the important users/contents from his/her perspective, summarize the contents by issuing any topical queries, or capture the overall picture of the relationship between the involved users and contents, and finally have a deep understanding about those mining results.

Therefore, different from most of the traditional work, which targets the single aspect of the OSN data (either the user, the content, or the other aspect), this dissertation aims at designing a comprehensive and versatile framework to mine the given OSN data, and covering different aspects of the data, i.e., finding out "influential" users, summarizing the important and trending contents, and uncovering the relationship between the users and contents.

The benefits provided by this framework are significant due to the massive requirements from real-world applications. Let us have a look at some example applications. First, people may have to identify "important" users based on different standards. For example, people, who have some domain specific (e.g., machine learning) questions may want to seek help from experts in OSN, because those experts are "important" to them; marketers would like to target users, who can

help promote their brands' products in OSN, thus, those users are "important" in their eyes. Second, no one could read every piece of the content in OSN within limited time. Instead, most people would prefer receiving the general summarized updates from OSN, while keeping a close eye on his/her interesting topics. Third, some people may be curious about who posts a hot message in OSN, or what an "influential" user said in the OSN today. For example, when an individual receives a retweet from Twitter, stating that an earthquake is shaking Japan, he/she may wonder who posted this tweet at the very first, where is he/she? For another example, when an individual wants to acquire the opinion about the current job market, he/she may go to the Twitter system to "follow" Barack Obama to see his thoughts. Behind these examples, people may wonder what is the real force pulling specific users and contents together.

Besides the benefits for real world applications, this dissertation is significant because of its capability of improving the existing methods for mining the OSN data. Specifically, for influential user identification, this dissertation will not only find out who is important currently, but also predict who will be important in the future dynamically and continuously. For summarizing the contents, this dissertation is one of a few pioneers, which modifies and applies the traditional summarization techniques for the social network documents, e.g., posts, blogs, reviews, etc., and proposes a new summarization method — storyline generation to provide more meaningful summaries for the time-sensitive social network documents, such as Tweets. After that, people could save time by reading the summaries of the news and updates on the OSN. In addition, two other methods, including an event detection framework and a multi-task multi-label classification method, are used to summarize the Time-Sensitive OSN contents. For organizing the user and content, this dissertation explores two directions: first, it aims at organizing the user and content of the OSN into a tree structure; second, it considers incorporating one type of object into the analysis of the other type of object to reveal the

influence of one to the other. For example, the instance-level constraints of one type of object (e.g., two objects must/cannot be placed together in the same node of the resulting tree) can be incorporated into the re-organization of the other type of object.

1.4 Contributions

This dissertation addresses three aforementioned research problems related to users and user-generated contents in OSN, using data mining and information retrieval techniques. It attempts to discover interesting patterns, summarize the historical records, and predict futures of the online social network.

1.4.1 Organization of Users and Contents

This dissertation presents a new way to organize the users and user-generated contents on the social network [LL10, LLO10, LSLO12]. Specifically, a Hierarchical Co-Clustering algorithm is introduced to organize the users and contents into a tree structure. This resulting tree can help with user/content retrieval and recommendation in the future. This algorithm can be applied to various types of OSN data sets, e.g., users and topics mentioned by those users, artists and textual labels assigned to those artists, etc. Based on the tree structure derived from our clustering algorithm, some pre-defined instance-level constraints can be incorporated into our method, and lead to better clustering performance, or organization performance.

1.4.2 Summarization of Contents

Aiming at delivering succinct summaries of the social network contents to the online audience, this part of dissertation presents a multi-purpose summarization framework [LLL11, LLL12a] as well as a novel summarization method – storyline generation [LLL$^+$12b]. Furthermore, two nontraditional methods, including an

event detection framework and a multi-task multi-label classification method, are proposed as alternatives.

The multi-purpose summarization framework is based on various Submodular Functions, which are adopted from a set function in Mathematics. This summarization framework performs four summarization tasks, including Generic Summarization (summarize a given set of documents), Query-Focused Summarization (given a query, summarize the contents related to the query), Update Summarization (given an existing summary regarding a topic, summarize the latest updates of this topic), and Comparative Summarization (given a query, summarize two to-be-compared document sets over multiple aspects of the query).

The storyline generation provides a new way to understand the time-sensitive textual contents from the social network. Different from the traditional summarization methods, which only try to extract core pieces of information, and then use them to form a summary without considering the chronological order of the events described by those information, the storyline generation pays special attention to the chronological order. It makes sense since (1) OSN is usually considered to be a novel type of media for news acquisition, and the time is quite important in this case; and (2) for a story described by the social network contents, "one" single cause might be followed by "multiple" effects, and the chronological order of events in the story can reveal the big picture of those "causes-effects" relationships.

Besides the traditional summarization methods, two alternatives are proposed to address some OSN audience's requirements. First of all, an event detection framework is designed to capture the "events" happening in the social network. This is especially useful for the audience, who are only interested in receiving the news/updates in a timely fashion without too much descriptions of them. Second, a multi-task multi-label classification method is introduced to help audience quickly classify a new coming OSN message into a topic and a sentiment.

1.4.3 Identifying Influential Users

This dissertation presents three important dimensions of social influence, including (1) Monomorphism vs. Polymorphism; (2) High Latency vs. Low Latency; and (3) Information Inventor vs. Information Spreader. They help with understanding the characteristics of "influential users" obtained from various different methods. The work about these three dimensions sheds light on the selection of appropriate methods for identifying influential users under specific circumstances. It has been accepted by the journal — Expert System With Applications.

In addition to the three dimensions of social influence, this dissertation proposes a novel dynamic influence model based on Continuous-Time Markov process to identify influential users according to the number of adopters (who follows the influential user candidate) [LPLS13]. This model can dynamically predict the influence of users. Here, "dynamically" means that given any time point in the future, this influence model can predict the user's influence at that specific time point.

1.5 Chapter Organization

The remainder of the dissertation is organized as follows. Chapter 2 introduces the new method for organizing users and contents in the social network. Chapter 3 and 4 describes the multi-purpose summarization framework for the social network textual contents, the storyline generation method, the event detection framework, as well as the multi-task multi-label classification method for the time-sensitive social network contents, e.g., Tweets from Twitter [1]. Chapter 5 presents three "dimensions" of the influence, and the novel dynamic influence model for identifying influential users in the social network. Finally Chap 6 concludes the dissertation.

[1] https://twitter.com/

CHAPTER 2

ORGANIZATION OF USERS AND CONTENTS

In the social network information retrieval, an important research topic, which has attracted much attention recently, is the utilization of user-generated contents, such as the topics, tags, keywords, and other textual labels, which can be extracted from the online social network web sites, e.g., Facebook [1], Twitter [2], Flickr [3], Last.fm [4], Youtube [5]. A fundamental research problem in the area is how to understand the relationships among users (in the OSN) and these different pieces of information, and then utilize the relationships to organize these two types of data together so that the future retrieval can benefit from the organization results.

Clustering algorithms provide clusters of data points, and it can be considered as a way to organize the data. Co-clustering is the problem of simultaneously clustering two types of data (e.g., documents and words, and webpages and urls). We can naturally bring this idea to the situation at hand and consider clustering users and topics together, users and tags together, or users and keywords together.Once such co-clustering has been successfully completed, one can identify co-existing clusters of users and topics, tags, or keywords [6].

When dealing with tags, it is worth noticing that some contents are more specific versions of others. This naturally suggests that the contents could be organized in hierarchical clusters. Such hierarchical organizations exist for topics and keywords, so we will consider hierarchical co-clustering of uses and contents.

---

[1] https://www.facebook.com/

[2] https://twitter.com/

[3] https://www.flickr.com/

[4] http://www.last.fm/

[5] http://www.youtube.com/

[6] Topics, tags, and keywords are all textual contents. For simplicity, we use contents to refer to topics, tags, or keywords for the rest of the dissertation when the organization of users and topics, tags, or keywords is being discussed.

In this dissertation, we systematically study the application of *H*ierarchical *C*o-*C*lustering (HCC) methods for organizing the social network data. There are two standard strategies for hierarchical clustering. One is the divisive strategy, in which we attempt to divide the input data set into smaller groups recursively, and the other is the agglomerative strategy, in which we attempt to combine initially individually separated data points into larger groups by finding the most closely related pair at each iteration. We will compare these two strategies against each other. We apply a previously known divisive hierarchical co-clustering method and a novel agglomerative hierarchical co-clustering. In addition, we demonstrate that these two methods have the capability of incorporating instance-level constraints to achieve better performance. We perform experiments to show that these two hierarchical co-clustering methods can be effectively deployed for organizing the music social network data and they present reasonable clustering performance comparing with the other clustering methods. A case study is also conducted to show that HCC can be applied for more other applications, such as quantify the similarity between social network users.

2.1 Overview

The traditional social network information retrieval research is mainly concerned with the users in the social network. Specifically, users and user-user relationship together are considered as a graph structure, and the graph algorithms are utilized to retrieve the users.

More recently, the user-generated information is brought to this research area. This information can help (1) summarize the profile of users; (2) recommend specific information to the particular group of users.

What has made such research possible is the increase of social-networking web sites in which users are permitted to post their current status, blogs, tweets, music, videos, pictures, etc., leave comments about the contents they read, listened to,

or watched in the form of short comments, and read comments of other users. In the comments, a wide variety of categorical information about contents and users are available to help users to make quick selection of contents to read, listen to, or watch. Let us raise the music social network as an example. Artists and listeners in the music social-networking websites might communicate with each other in the form of comments about music tracks, and the music tag, style, or mood related to the comments shows the categorical information of music tracks.

We are particularly interested in these additional categorical information since (1) they are commonly available in nearly all of social network web sites in the form of topics, tags, or keywords; (2) they are reasonable features to compute the similarity between posts (i.e., blogs, tweets, music, images, videos). By sampling representative posts of an author/user, one is able to gather topics, tags, and keywords of this author/user.

An important characteristic of these contents [7] is that some contents are more general while some others are more specific. For the example of music social network, e.g., "Soft Metal" is a more specific style (style is a kind of keywords in the music social network) than"Metal", "Dance Pop" is more specific than "Pop", "Extremely Provocative" is a more specific tag than "Provocative", and "Agony" is a more specific mood label (mood is another kind of keywords in the music social network) than "Sadness".

Since there is no limit in the length of any of the contents of data, a topic, a tag or a keyword can be an extension of another topic, tag or keyword accordingly, which is an extension of yet another topic, tag or keyword. This suggests that the contents cannot only be clustered into a one-level clustering structure but also a hierarchical clustering structure.

Hierarchical clustering is the problem of organizing data in a tree-like structure in which the input set of data points is recursively divided into smaller subgroups,

[7]It may be somewhat redundant to call them "tag labels" but for simplicity we view all of them as "contents".

usually until the subgroups become individual data points. Hierarchical clustering offers natural facilitation of data navigation and browsing [CKPT92], it has been studied quite extensively in bioinformatics [GLD00, ESBB98], in image analysis [Pha01, BF01], and document analysis [XLG03, ZKF05, FWE03]. There are two standard strategies for hierarchical clustering. One is the divisive strategy, in which we attempt to divide the input data set into smaller groups recursively, and the other is the agglomerative strategy, in which we attempt to combine initially individually separated data points into larger groups by finding the most closely related pair at each iteration. Co-clustering is the problem of developing organizations of two or more types of data. Much less research has been done on co-clustering than on hierarchial clustering [CL04]. While both hierarchical clustering and co-clustering have their own advantages, few algorithms exist that execute both simultaneously [Ber06].

We assume that, given a set of representative features, hierarchically organizing user individually or contents can be effectively accomplished by computation. So we question whether attempting to cluster two data types together will lead to better organizations, which we will study in the dissertation.

In this dissertation, we systematically study the application of hierarchical co-clustering (HCC) methods for organizing users and contents (textual labels of topics, tags, or keywords). We first examine the hierarchical divisive co-clustering algorithms [XM06]. This algorithm has already been incorporated into the framework for quantifying artist similarity and is capable of generating reasonable double-hierarchies [SLO08].

After experimenting with a recently proposed hierarchical divisive co-clustering method, HDCC (Hierarchial Divisive Co-Clustering), we present a novel method, HACC (Hierarchical Agglomerative Co-Clustering). The divisive HDCC combines Singular Value Composition (SVD) and K-means using a top-down iterative process. The agglomerative HACC starts with singleton clusters and then repeatedly

merging two nearest clusters into one until all the points are merged into one cluster. These two methods share a special characteristic: *grouping points from both data types*. In the case of HDCC, this means that during each "divisive" step, the users are split into different clusters, while at the same time the content labels are split into corresponding clusters containing appropriate users. In the case of HACC, this means that at each step of the merging process, HACC can merge a subset of the users with a subset of the content labels based on their internal heterogeneity. In practice, when our goal is to build double-hierarchies for users and contents, one can often observe that a group of users and a group of content labels are exclusively correlated with each other (i.e., not correlated with any other users or contents). HACC aims at, in such a situation, merging them into a single group at the earliest possible stage [LLO10, EO93, MAH95].

Our hope is that such clusters with two data types will be used for (1) better retrieval when both types of data are specified in a query, e.g., given a query including a user and one of his/her topic labels, one can probably retrieve them together from a user-topic hierarchy, while with the query composed of a user and a tag, one can retrieve them simultaneously from a user-tag hierarchy; (2) the recommendation application when one wishes to find out if a content label is suitable for a user and this user has never commented or been commented with the that content label. E.g., given a query including a user and a topic label this user has not mentioned before, one can retrieve them together from a user-topic hierarchy. If they can be found in the same cluster, this topic label will be recommended to the user.

Figure 2.1 shows a sample output dendrogram of a traditional hierarchical clustering method and Figure 2.2 shows a sample output dendrogram of HCC. In this dissertation, we show that such mixed-data-type hierarchical clusters can be generated by HCC and empirically better clusters generated by concurrent use of two data types. We also show that HCC can be extended to incorporate instance-

	Artist
A1	Air
A2	Nine Inch Nails
A3	Radiohead
A4	Metallica
A5	Led Zeppelin
A6	Nirvana
A7	Pink Floyd
A8	Soundgarden
A9	Massive Attack
A10	The Beatles

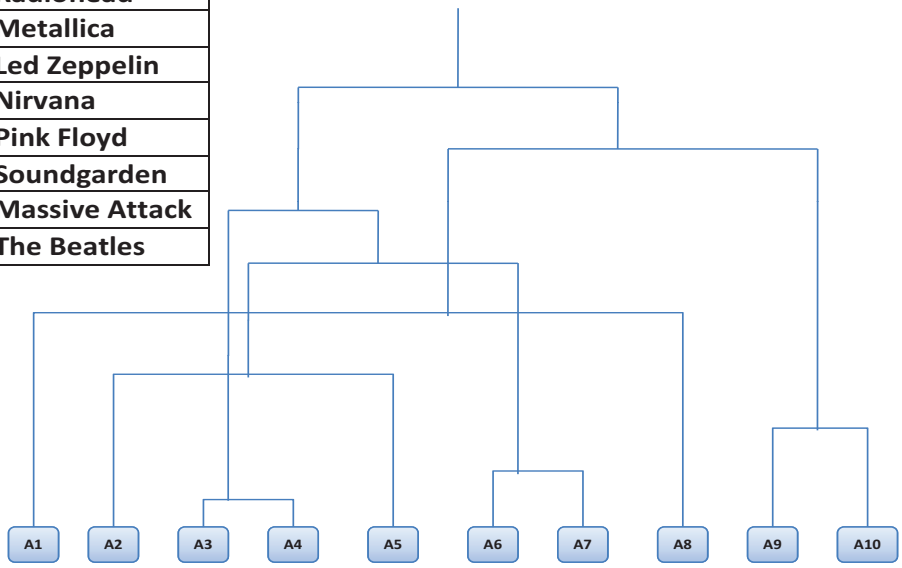


Figure 2.1: Part of the dendrogram generated by SLHC.

level constraints that specify certain content labels must be or must not be together or certain users must be or must not be together for better organization.

Our contributions in this part of the dissertation are three-fold: 1) we develop a novel hierarchical co-clustering method to organize the social network data and facilitate the retrieval given a query including two types of data; 2) we incorporate the instance-level constraints into HCC method and show that such constraints-incorporated HCC could provide better clustering performance; 3) we perform a case study to show that HCC methods have the capability of providing reasonable user similarity quantification measures.

2.2 Related Work

Hierarchical Clustering is the generation of tree-like cluster structures without user supervision. Hierarchical clustering algorithms organize input data either bottom-up (agglomerative) or top-down (divisive) [TSK+06]. In general hierarchical agglomerative clustering is more frequently used than hierarchical divi-

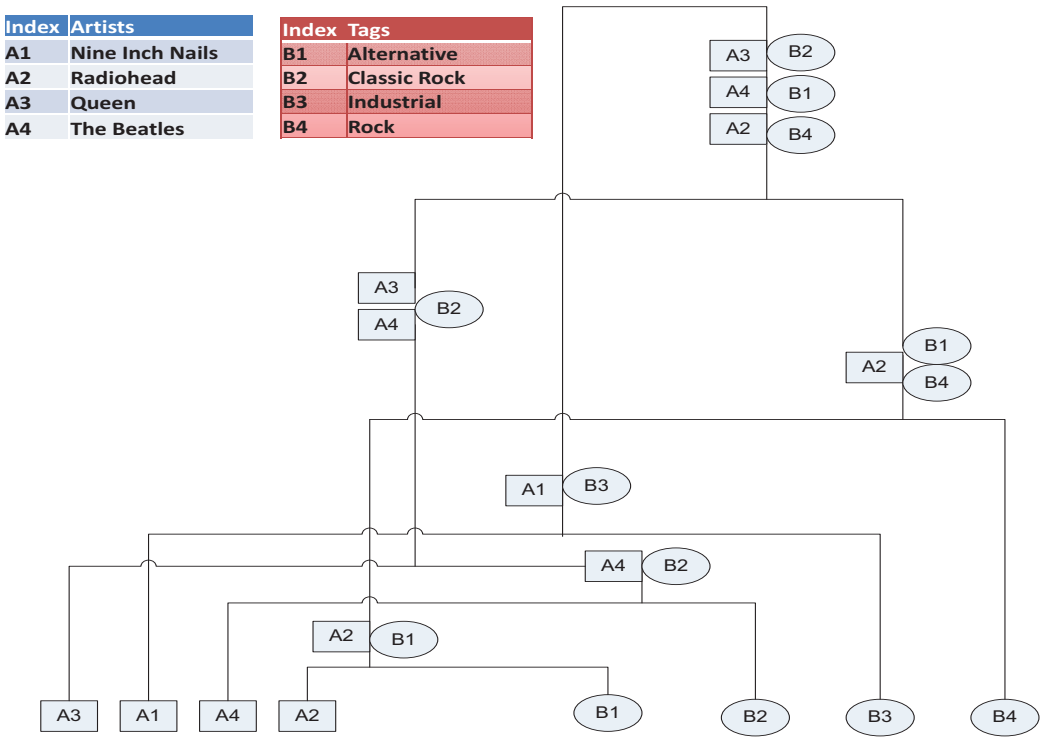


Figure 2.2: Part of HCC dendrogram. Rectangles represent artists and ellipses represent tags assigned to these artists. The nodes containing both rectangles and ellipses are clusters containing both artists and tags.

sive clustering. **Co-clustering** refers to clustering of more than one data type. Dhillon [Dhi01] proposes bipartite spectral graph partitioning approaches to co-cluster words and documents. Long et al. [LWZY06] propose a general principled model, called Relation Summary Network, for co-clustering heterogeneous data presented as a k-partite graph.

While hierarchical clustering deals with only one type of data and co-clustering produces only one level of data organization, **hierarchical co-clustering** aims at simultaneously constructing hierarchical structures for two or more data types, that is, it attempts to achieve the function of both hierarchial clustering and co-clustering. Because of this unique nature hierarchical co-clustering is receiving special attention from researchers [HA07, IPM09]. Xu et al. proposed a **hierarchical divisive co-clustering** algorithm [XM06] to simultaneously find out document clusters and the associated word clusters. Shao et al. [SLO08] incorporated this hierarchical divisive co-clustering algorithm into a novel artist similarity

quantifying framework for the purpose of assisting artist similarity quantification by utilizing the style and mood clusters information. In their framework, the artist similarity is based on style similarity and mood similarity. Even though this hierarchical divisive co-clustering method is given, to our best knowledge, few researchers have studied the hierarchical agglomerative co-clustering methods (e.g., Li et al. [LL10] made the initial attempt to study a hierarchical agglomerative co-clustering method).

In recent years much work has been done on **constrained clustering** — integrating various forms of background knowledge in the clustering process. Existing constrained clustering methods have been focused on the use of background information in the form of instance level "must-link" and "cannot-link" constraints, which, as the naming suggests, assert that, for a pair of data instances, they must be in the same cluster and they should be in distinct clusters, respectively. Most of constrained clustering algorithms in the literature are designed for partitional clustering methods (e.g, constrained K-means clustering, constrained spectral clustering, and constrained clustering using non-negative matrix factorizations, see a survey [BDW08]) and little has been done on utilizing constraints for hierarchical clustering. Recently, there do exist a few works on incorporating constraints into hierarchical clustering (e.g., by extending the partial known hierarchy with the constraints to a full hierarchy or by modifying the order of cluster merging process) [BN08, GD11, DR09, ZL11]. However, these constrained hierarchical clustering methods cannot be applied to our hierarchical co-clustering problem. In our settings, the effects of constraints over one type of data can be transferred to the other type of data, so that both types of clustering would benefit from them.

2.3 Hierarchical Co-Clustering Methods

We begin this section by describing the details of our application of the hierarchical divisive co-clustering algorithm (HDCC) by Xu et al. [XM06] to the problem of co-clustering user-content. The procedure is similar with that in Shao et al. [SLO08]. We then present a novel hierarchical agglomerative co-clustering algorithm called HACC, which could also be utilized to cluster user-content.We will compare this agglomerative method with the previous divisive method based on their clustering performance.

2.3.1 Problem Formulation

Suppose we are given a set of m users $\boldsymbol{A}=\{a_1, a_2, \ldots, a_m\}$, and a set of n unique contents that are assigned to these users, $\boldsymbol{T}=\{t_1, t_2, \ldots, t_n\}$. Suppose we are also given an $m \times n$ user-content relationship matrix $X = (x_{ij}) \in \mathbb{R}^{m \times n}$, such that x_{ij} represents the relationship between the i-th user in $\boldsymbol{A}$ and the j-th content in $\boldsymbol{T}$ (e.g., x_{ij} can be the frequency of the assignments of the j-th content to i-th user). Our goal is to simultaneously generate a hierarchical clustering of $\boldsymbol{A}$ and of $\boldsymbol{T}$ based on matrix X so that each user and content can be in the appropriate cluster and show the hierarchical relationships of these clusters.

2.3.2 Hierarchical Divisive Co-Clustering

We first directly apply the hierarchical divisive co-clustering [XM06] to generate a user-content hierarchy.

In this application, the user and content together is represented as an user–content matrix X. As we mentioned before, content can be any of the aspects of social network users, i.e., topics, tag labels, keyword labels, and etc. The key idea behind the method is to combine Singular Value Decomposition (SVD) (gives us the partitioning of users as well as the partitioning of contents at the same time)

and K-means (provides us the optimal bipartitioning of users and contents) using a top-down iterative process [XM06]. The procedure is described as follows:

1. Given an $m \times n$ user–content matrix, X, perform SVD decomposition on X to obtain X_n:

$$D_1 = diag([1]_{1\times m}X), D_2 = diag(X[1]_{n\times 1})$$

$$X_n = D_1^{-1/2} \times X \times D_2^{-1/2}$$

$$X_n = U \times \Lambda \times V^T$$

2. Let $\lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_m$ be the largest m singular values. Then the number of clusters is k where:

$$k = \text{argmax}_{(m\geqslant i>1)}(\lambda_{i-1} - \lambda_i)/\lambda_{i-1}$$

3. Find k singular vectors of X_n: $u_1, u_2, \ldots, u_k$ and $v_1, v_2, \ldots, v_k$, and then form a matrix Z by:

$$Z = \begin{bmatrix} D_1^{-1/2}[u_1, ..., u_k] \\ D_2^{-1/2}[v_1, ..., v_k] \end{bmatrix}$$

4. Apply K-means clustering algorithm to cluster Z into k clusters. Note that the first m labels belong to users, while the rest n belong to contents.

5. For each cluster which contains both users and contents, we check the number of users in it. If the number is higher than a given threshold (in our experiment, we set the threshold = 3), construct a new user–content matrix formed by the user and contents in that cluster, and continue to the first step.

Figure 2.3 is a sample cluster obtained from the artist-style (both artists and style labels are extracted from music social network web sites) dendrogram. In this cluster, we observe that the pair of *Country-Rock* and *Progressive Country*, and the triple of *Americana*, *Alternative Country* and *Neo-Traditional Folk* are probably the most similar (They can be grouped together in the top layer) in style description, and the similarity between *Country-Pop* and *Urban Cowboy* is greater than the similarity between *Country-Pop* and *Cajun* as well as the similarity

Figure 2.3: A sample cluster content from user-keyword (artist-style) hierarchy. (=means the most similar).

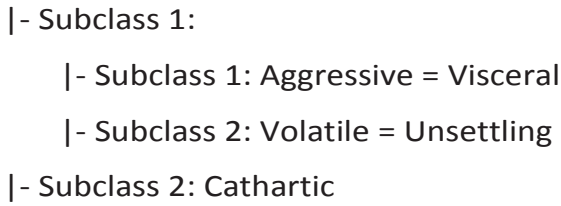


Figure 2.4: Another sample cluster content from user-keyword (artist-mood) hierarchy. (=means the most similar).

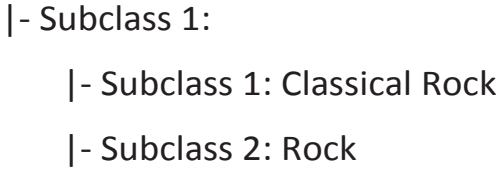


Figure 2.5: A sample cluster content from user-tag (artist-tag) hierarchy. (=means the most similar).

between *Urban Cowboy* and *Cajun*. Figure 2.4 is a sample cluster obtained from the artist-mood (both artists and mood labels are extracted from music social network web sites) dendrogram following the same construction rule. Similarly, Figure 2.5 is a sample cluster obtained from the artist-tag (both artists and tag labels are extracted from music social network web sites) dendrogram.

Based on this hierarchical divisive co-clustering algorithm, we can also obtain the tag-based artist clusters, style-based artist clusters as well as mood-based artist clusters. They have the similar well-balanced cluster member distributions.

2.3.3 Hierarchical Agglomerative Co-Clustering

Here we present our novel hierarchical agglomerative co-clustering (HACC) algorithm. Like the traditional agglomerative hierarchical clustering algorithms, HACC starts with singleton clusters and then successively merges the two nearest clusters until only one cluster is left. However, unlike traditional algorithms, it may unify classes from two different data types. This means that the cluster left at the end consists of all the rows and columns and so if there are m rows and n columns exist, HACC executes $m + n - 1$ rounds. The output of HACC is thus a single tree where the leaves are the rows and the columns of the input matrix, where nodes having both rows and columns as descendants may appear at any non-leaf level. Figure 2.2 illustrates a dendrogram example generated by HACC.

The algorithm of HACC is presented in Algorithm 1. The central part in the design of Algorithm 1 is the method PickUpTwoNodes, which is for selecting two nodes (corresponding to two clusters) to merge. For the purpose of creating groups consisting of two different data types, we use cluster heterogeneity measurement, denoted by CH. Given a group C consisting of r rows, P, and s columns, Q, we define $CH(C)$ as

$$CH(C) = \frac{1}{rs} \sum_{i \in P, j \in Q} (x_{ij} - \mu)^2, \tag{2.1}$$

Algorithm 1 HACC Algorithm Description

Given A – the set of the first type of data points (e.g., artists) and B – the set of the second type of data points (e.g., T/S/Ms).
Create an empty hierarchy H
$List \leftarrow Objects\ in\ A + Objects\ in\ B$
$N \leftarrow size[A] + size[B]$
Add $List$ to H as the bottom layer

for $i = 0$ to $N - 1$ **do**
  p, q = PickUpTwoNodes($List$)
  o = Merge(p, q)
  Remove p, q from $List$ and add o to $List$
  Add $List$ to H as the next layer
end for

where μ is the average of entries over rows P and columns Q; i.e., $\mu = \frac{1}{rs} \sum_{i \in P, j \in Q} x_{ij}$. CH measures the local proximity of the cluster. It is worth noticing that CH indicates how "close" the users and the contents involved in the cluster are, specifically speaking, the lower CH means the users and the contents are closer. The goal of choosing two nodes to merge is to let the users and the contents within the resulted cluster as close as possible. As a result, for a merger, we choose two nodes whose merging would result in the least increase in the total cluster heterogeneity [EO93].

Similarly as before, we can also obtain the user/content clusters from the user-content dendrogram generated by HACC.

2.4 Incorporating Instance-level Constraints for HCC

In practice, one may observe pairs of users that should be clustered into the same cluster. Similarly, one may observe pairs of contents that must be always in the same content cluster. These observations are represented as the aforementioned "must-link" and "cannot-link" constraints. We design hierarchical co-clustering (HCC, including HACC and HDCC) to incorporate such constraints.

There are two issues in incorporating these constraints. One is how to use them for grouping data points of the same type; i.e., how to use user constraints for grouping users and content constraints for grouping contents. The other is how

to transfer constraints on one data type to the other data type. To address the first issue, we use Dunn's Index to determine the best layer for cutting the HCC-generated dendrogram and then apply the constrained K-means to incorporate the constraints of the same data type. To address the second issue, we use an alternating exchange algorithm.

2.4.1 Best Layer

Since HCC produces a list of clustering results and each clustering corresponds to one layer of the dendrogram, we use Dunn's Validity Index [Dun73] to measure and compare these clusterings. This validity measure is based on the idea that good clustering produces well-separated compact clusters. Given a clustering layer consisting of r clusters $c_1, \ldots, c_r$, Dunn's Index is given by:

$$D = \frac{\min_{1 \leq i < j \leq r} d(c_i, c_j)}{\max_{1 \leq k \leq r} d'_k}, \tag{2.2}$$

where $d(c_i, c_j)$ is the inter-cluster distance between the i-th and the j-th clusters and d'_k is the intra-cluster distance of the k-th cluster. Generally, the larger Dunn's Index, the better the clustering.

After determining the best layer to cut the dendrogram, we can easily make use of the constraints of the same data type. In particular, we perform constrained K-means on the best layer with the parameter K set to the number of clusters in that layer. For this purpose, we use the MPCK-Means algorithm in [BBM04]. In general, MPCK-Means incorporates both metric learning and the pairwise constraints effectively into semi-supervised clustering. It adopts EM framework to perform cluster assignment in the E-step, and centroid estimation and metric learning in the M-step. Notice that the constrained K-means is applied on a single data type, afterwards, the constraints can be transferred to the other data types to improve their clusterings.

2.4.2 Alternating Exchange

Here we show how to transfer the constraints between different data types. Specifically, at the best layer of the dendrogram generated by HCC, if some user (or content) data points of certain node are being re-assigned to another node at the same layer after using the instance-level constraints, we can use an alternating exchange algorithm [GS96] to improve T/S/M (or artist) clustering.

The objective function of clustering can be written as [GS96]:

$$Z = \sum_{k=1}^{r} \sum_{l=1}^{m} \sum_{i \in A_k} \sum_{j \in T_l} (x_{ij} - w_{kl})^2, \tag{2.3}$$

with

$$w_{kl} = \frac{1}{a_k t_l} \sum_{i \in A_k} \sum_{j \in T_l} x_{ij}. \tag{2.4}$$

Here r is the number of type A (represents users) clusters, m is the number of type T (represents contents) clusters, A_k is the k-th cluster containing data points of type A, T_l is the l-th cluster containing data points of type T, a_k and t_l respectively denote data points of type A and T. As before, x_{ij} is the value representing the relationship between the i-th type-A data point and the j-th type-T data point.

To transfer constraints from contents to users, we do the following: Suppose we have just obtained a clustering of users, C_A, and a clustering of contents, C_T, by cutting the HCC dendrogram using Dunn's index, as described before. We first incorporate into these clusterings the content constraints using the techniques described in Section 2.4.1 thereby obtain an improved content clustering, C'_T. Then we execute the greedy algorithm shown in Algorithm 2 to make changes on user class assignments. The greedy algorithm is aimed at minimizing the quantity Z in (2.3) and in each round one user is moved from the current cluster to another if that move decreases the value of Z. Transferring constraints backward (i.e., from artists to T/S/Ms) could be done by simply switching the role of contents and users. In our implementation, we transfer only from contents to users.

Algorithm 2 Alternating Exchange Algorithm

Input: clusterings C_A and C'_T, and normalized A-T matrix X, where C'_T is obtained by using the MPCK-Means on the output of HCC with respect to Content constraints.

while There is an user whose relocation from the current cluster to another decreases the value of Z **do**
  pick an user-destination pair that maximizes the decrease and relocate the user to the destination
end while
Output the resulting user clustering C'_A

2.5 Experiment

Here we present results of our experiment. For the sake of presentation, we will only show a part of results on the user-content hierarchy. From the representative part shown here, one can imagine the overall picture of the resulted hierarchy.

To further show the advantages of our HCC method, we will use the music social network data to make a case study of artist similarity quantification to compare the hierarchy generated using our method and the hierarchy generated using timbral features along with wavelet coefficient histograms.

2.5.1 Data Set

Music social network web sites enable the users to assign tags, style keywords, and mood keywords to their interested music tracks. Due to the fact that music tracks are closely related to the artists (they create those works) and the listeners (they listen to, "favorite", or share those works), it is natural to treat the related tags, style keywords, and mood keywords as an important kind of contents for those artists and users in the music social network service. Thus, artist/user and tag/style/mood [8] can be considered as a typical example of "user and content" in the music social network.

[8]The acronym "T/S/M" is used to repent these contents in the music social network.

We use the music social network data set in [WWS+09] consisting of 403 artists. For each artist, contents, including tags and styles are collected from Last.fm (http://www.last.fm). There are 8,529 unique tags and 358 unique style keywords. Concerning about the mood information, we collect all the mood keywords for the 403 artists from All music guide (http://www.allmusic.com). Note that an artist may receive the same tag more than once, while is assigned the same style/mood only once. By counting the number of assignments by the same tag, each artist is represented by a 8,529-dimensional integer vector. We scale these tag vectors so that the total of the 8,529 entries is equal to a fixed constant. We will use X to denote the artist-tag frequency matrix thus generated.

As to the style keywords, each artist belongs to at least one style and each style contains at least one artist. We generate an artist-style incident matrix from the data, so that the entry at coordinate (i, j) is 1 if the i-th artist has the j-th style label and 0 otherwise. Similarly, we generate an artist-mood incident matrix from the data.

For the following experiments, we focus on hierarchical co-clustering of artist and tag data, while the empirical studies for the artist and style/mood are omitted because they show similar results.

2.5.2 Hierarchies Generated from HDCC

We use HDCC to generate dendrograms of the artists and tags, artists and style keywords and artists and mood keywords. For the case of artist-T/S/M dendrogram, originally, all the artists and T/S/Ms are within the same cluster. As HDCC running, the artists and T/S/Ms are iteratively separated into different clusters from the higher layer to the lower layer until the number of artists within each cluster of the lower layer is not larger than the given threshold. A sample part of the dendrogram generated by HDCC is presented in Figure 2.6.

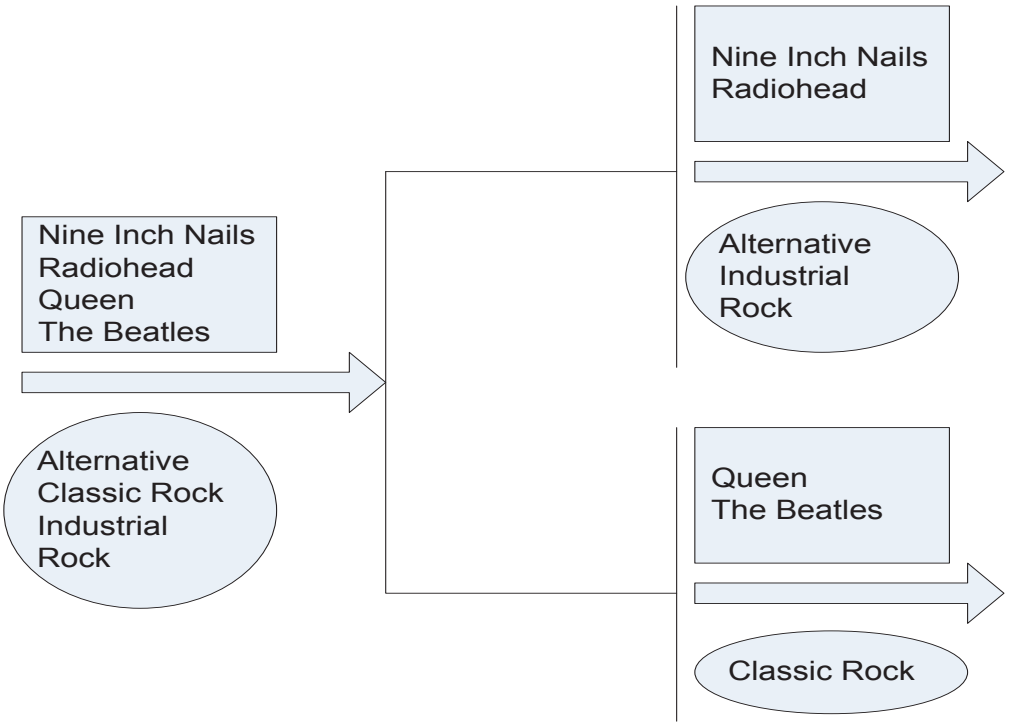


Figure 2.6: Part of HDCC dendrogram. It shares the same artists and tags as Figure 2.2.

2.5.3 Hierarchies Generated from HACC

We use unconstrained HACC (no instance level constraints are given in this experiment) to generate dendrogram of the artists and the T/S/Ms. Figure 2.2 is part of the dendrogram generated by HACC in our experiment. In the dendrogram, each leaf represents one artist or one tag, each internal node contains a subset of artists and tags, and the top layer is the cluster that contains all artists and tags. Because many people assign a tag "Industrial" to artist *Nine Inch Nails*, "Industrial" and *Nine Inch Nails* are clustered together. The novelty here is that artists and tags are jointly organized into a hierarchical structure. Once such a hierarchical organization has been generated, an artist can be described by the tags that appear in its cluster. The more representative are the tags for certain artists, the larger possibility for them to be clustered together.

2.5.4 Clusterings Comparison

Hierarchical Clustering Performance Comparison

We compare the HDCC-generated and HACC-generated dendrograms with one generated by single linkage hierarchical clustering (SLHC). It is worth noticing that both HACC and HDCC algorithms used for hierarchical clustering performance comparison do not consider must-link or cannot-link constraints. SLHC is the standard hierarchical clustering method and thus serves as our baseline. Since SLHC can cluster only one type of data, we provide SLHC with the normalized artist-tag matrix by viewing each row as the feature vector of the corresponding artist and produce hierarchical clustering of artists. The artist dendrogram generated by SLHC is shown in Figure 2.1.

To evaluate and compare these three artist dendrograms, we utilize CoPhenetic Correlation Coefficient (CPCC) [SR62] as evaluation measure. Intuitively CPCC measures how faithfully a dendrogram preserves the pairwise distances between the original data points. CoPhenetic Correlation Coefficient (CPCC) is given as:

$$CPCC = \frac{\sum_{i<j}(d(i,j)-d)(t(i,j)-t)}{\sqrt{(\sum_{i<j}(d(i,j)-d)^2)(\sum_{i<j}(t(i,j)-t)^2)}} \tag{2.5}$$

Here $d(i,j)$ and $t(i,j)$ are respectively the ordinary Euclidean distance and the dendrogrammatic distance between the i-th and the j-th data points (e.g., artists), and d and t are their respective averages. The comparison results based on CPCC are shown in Figure 2.7, and so we conclude that HDCC cannot generate a "good" enough dendrogram comparing with SLHC while our HACC method generates the most faithful dendrogram in terms of hierarchical clustering performance. Through the coupled dendrogram generated either by HDCC or HACC, one can observe the relationship between artists and T/S/Ms, also make use of the T/S/Ms within the same cluster as some artists to explain why these artists are clustered together.

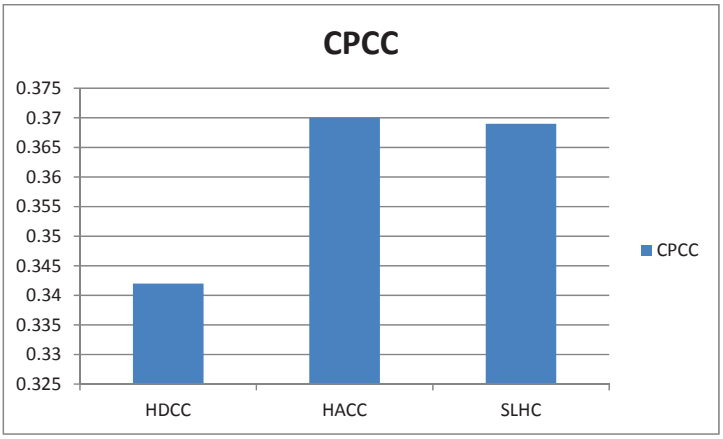


Figure 2.7: CPCC of HDCC, HACC and SLHC.

Flat Clustering Performance Comparisons

We also evaluate the artist clustering performance of HACC and HDCC, by comparing it with three co-clustering algorithms including Information-Theoretic Co-clustering (ITCC) [DMM03], Euclidean Co-clustering (ECC), and Minimum Residue Co-clustering (MRC) [CDGS04] on the artist-tag dataset.

Since the styles are assigned by some professional experts, we believe that some of the styles can be treated as the oracle class labels of the given artists.

We first cluster the styles using K-means clustering based on the artist-style matrix (that is, clustering of the columns, where each column is the 403-dimensional 0/1 vector that shows assignments of the style corresponding to the 403 artists). We then treat each cluster as a label and assign to each artist one label in the following manner:

- If all the styles assigned to an artist a belongs to a single cluster, we use that cluster as the label of a. Otherwise, choose the cluster with the largest number of styles assigned to a. If there is a tie, choose the one with the larger total number of styles, and if that doesn't break the tie, break it arbitrarily.

We use these labels as our ground truth class labels in the clustering performance measurements presented below.

Flat Clustering Evaluation Measures

We use Accuracy, Normalized Mutual information (NMI), Purity, and Adjusted Rand Index (ARI) as performance measures. These measures have been widely used in clustering evaluation and we hope they would provide insights on the performance of the HCC methods. For all these measures, the higher the value, the better the clustering.

Suppose we are given clusters $C_1, \ldots, C_k$ of size $c_1, \ldots, c_k$, respectively and we are comparing this clustering against the ground-truth clustering $E_1, \ldots, E_k$ of size $e_1, \ldots, e_k$. Let n be the total number of data points and for all i and j, let μ_{ij} denote the number of data points in both C_i and E_j.

Accuracy measures the extent to which each cluster contains the entities from corresponding class and is given by:

$$Accuracy = \max_{\pi} \frac{\sum_{i,\pi(i)} \mu_{i\pi(i)}}{n}, \tag{2.6}$$

where π ranges all permutations of $1, \ldots, k$. **Purity** measures the extent to which a cluster contains entities of a single class and is given by:

$$Purity = \frac{1}{n} \sum_{i=1}^{k} \mu_{i\rho(i)}, \tag{2.7}$$

where $\rho(i)$ is the j that maximizes μ_{ij}. **Adjusted Rand Index** is the corrected-for-chance version of Rand Index, and measures the similarity between two clusterings [MC86]. It is given by:

$$ARI = \frac{a - \frac{2bc}{n(n-1)}}{\frac{b+c}{2} - \frac{2bc}{n(n-1)}}. \tag{2.8}$$

Here $a = \sum_{i,j} \frac{\mu_{ij}(\mu_{ij}-1)}{2}$, $b = \sum_i \frac{c_i(c_i-1)}{2}$, and $c = \sum_j \frac{e_j(e_j-1)}{2}$. **NMI** is the normalized version of mutual information and measures how much information the two clusterings share [SG03] and is given by:

$$NMI = \frac{\sum_{i,j} \mu_{ij} \log(\frac{n\mu_{ij}}{c_i e_j})}{\sqrt{(\sum_i c_i \log \frac{c_i}{n})(\sum_j e_j \log \frac{e_j}{n})}}. \tag{2.9}$$

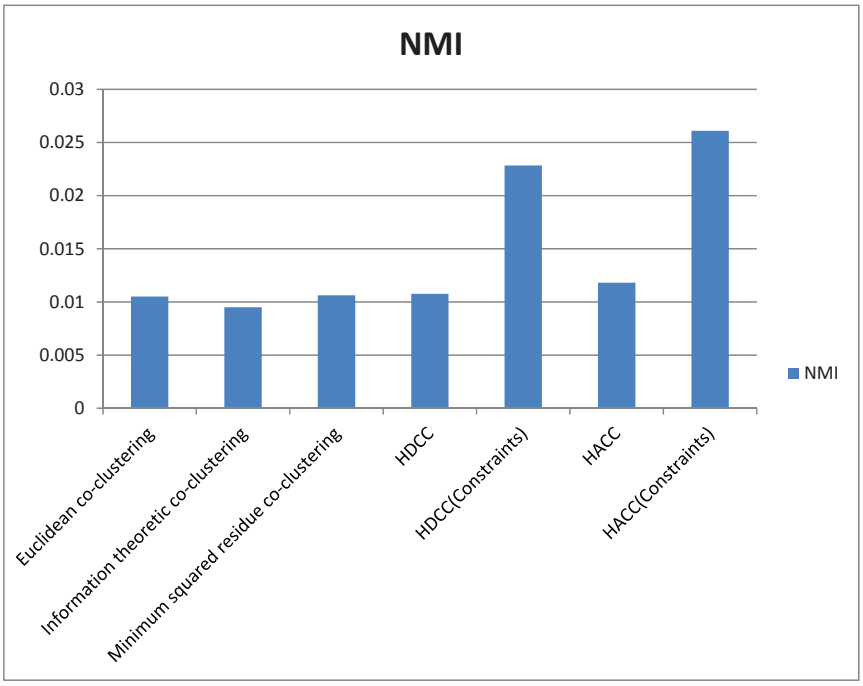


Figure 2.8: NMI of various clustering methods. HDCC(constraints) represents HDCC with 10 artist constraints. HACC(constraints) represents HACC with 10 artist constraints.

Flat Clustering Experimental Results

As we mentioned in Section 2.4.1, Dunn's Index can be used to find the best layer of the dendrograms generated by HDCC and HACC. After computing Dunn's Index on the clustering of each layer, we can find out the best layers of the two dendrograms generated by HACC and HDCC separately. Since we have already obtained the best layer, the clustering of this layer is compared against Co-clustering algorithms. This clustering is based on artist data points, we applied co-clustering algorithms for clustering artists.

Figure 2.8 shows the experiment results on the clustering methods using NMI as the performance measure. The results in the figures demonstrate that the HCC methods outperforms the co-clustering methods. Similar behaviors can be observed when using Accuracy, Purity, and ARI measures. Due to space limitation, we do not include the figures for Accuracy, Purity, or ARI. Figure 2.8 also shows that using the artist constraints improves the clustering performance.

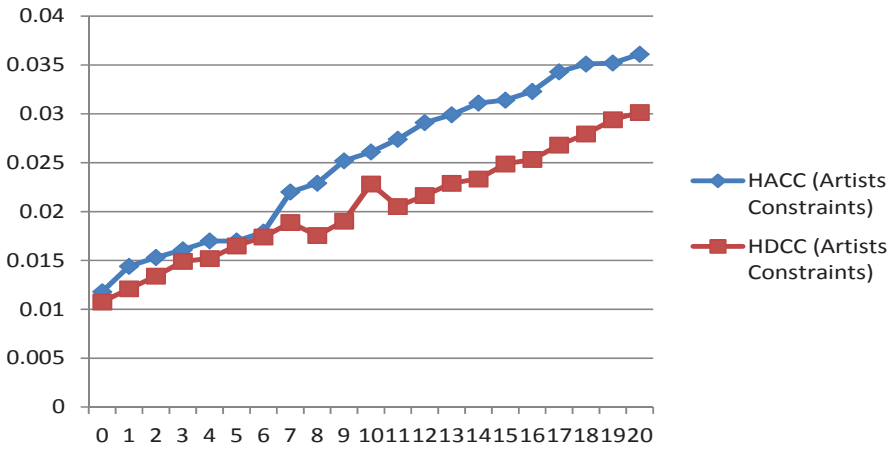


Figure 2.9: NMI on HACC with artists constraints range from 0 - 20.

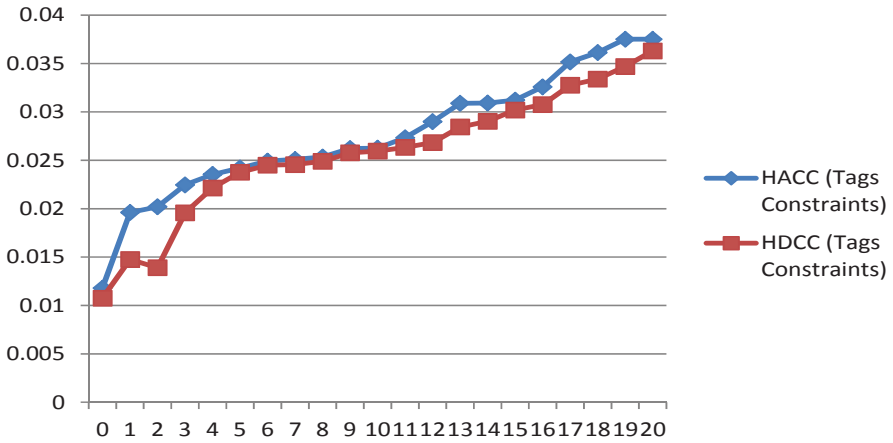


Figure 2.10: NMI on HACC with tags constraints range from 0 - 20.

We also evaluate NMI on HCC (including HACC and HDCC) with increasing number of constraints. The result in Figure 2.9 shows that the artist clustering performance improves with the increasing number of artist constraints. In other words, the artist constraints improves the clustering performance of HCC. Figure 2.10 shows that artist clustering performance improves as the number of tag constraints increases.

2.6 A Case Study

As the empirical results have shown that HCC methods can be utilized to organize the music social network artists and T/S/Ms as double-hierarchies and such double-hierarchies provide users a new way to do music social network in-

formation retrieval and reasonable clustering results, we conduct a case study for HCC methods to further show that the HCC methods could also be applied for computing the music social network artist/user similarity. As we have shown that HACC generate "better" hierarchy than HDCC according to CPCC and these two methods present similar clustering performance, it is believed that both can be employed to compute the artist similarity and show similar performance.

2.6.1 Similarity Quantification using HCC

We introduced the two hierarchial co-clustering algorithms. The first one (top-down) has already been successfully applied in a framework for quantifying musical artist similarity using style and mood information and shows very closely matchable performance compared to the artist similarity based on the acoustic features extracted from the related music [SLO08].

Notice that most of the music tags are assigned based on styles, genres, even the acoustic characteristics; while the music styles, genres and the other related information can reflect the distinguishing features of the artists. As we mentioned before, T/S/Ms show hierarchical relationships between each other, then one can find out the relationships among the artists based on the corresponding T/S/M information. Based on such idea, the first T/S/M can be an extension of the second one, which is an extension of the third one; same thing holds for the artists of these T/S/Ms. As a result, T/S/Ms can not only be hierarchically clustered but act as the important features of the related artists.

In [SLO08], only the styles and moods are used as features to quantify artist similarity. Here, we extend the framework in [SLO08] by adding tags as another important feature for quantifying the artist similarity and conduct a case study.

To calculate artist similarity, we need to quantify the semantic similarity between all triples of tag/style/mood terms first. In order to do this, we investigate the methods proposed by Resnik [Res95], Jiang and Conrath [JC97], Lin [Lin98],

and Schlicker et al. [SDRL06]. The approaches for calculating the similarity proposed by them are briefly described as follows:

Resnik:

$$sim_R(s_1, s_2) = \max_{s \in S(s_1,s_2)} \{-\log(p(s))\} \tag{2.10}$$

Jiang-Conrath:

$$\begin{aligned} & dist_{JC}(s_1, s_2) \\ = & \max_{s \in S(s_1,s_2)} \{2\log(p(s)) - \log(p(s_1)) - \log(p(s_2))\} \end{aligned} \tag{2.11}$$

Lin:

$$sim_L(s_1, s_2) = \max_{s \in S(s_1,s_2)} \{\frac{2 \times \log(p(s))}{\log(p(s_1)) + \log(p(s_2))}\} \tag{2.12}$$

Schlicker:

$$\begin{aligned} & sim_L(s_1, s_2) \\ = & \max_{s \in S(s_1,s_2)} \{\frac{2 \times \log(p(s))}{\log(p(s_1)) + \log(p(s_2))}(1 - \log(p(s)))\} \end{aligned} \tag{2.13}$$

Here $p(s) = freq(s)/N$ and $freq(s)$ is the number of artists those described by the given T/S/M term s, N is total number of artists, and $S(s_1, s_2)$ is the set of common subsumes of T/S/M terms s_1 and s_2. The basic idea of these approaches is to capture the specificity of each T/S/M term and to calculate the similarity between T/S/M terms that reflects their positions in the hierarchies generated by the methods presented in Section 2.3.2 and 2.3.3.

Once we obtain the pairwise semantic similarity of T/S/M terms, we can calculate the artist similarity based on T/S/M. For example, if artist A_1 is described by a group of styles $s_1, s_2, \ldots, s_i$, and artist A_2 is described by another group of styles $s'_1, s'_2, \ldots, s'_j$, we define the style-based similarity between A_1 and A_2 as:

$$sim(A_1, A_2) = \frac{\sum_{x \in [1,i]} (\max_{y \in [1,j]} sim(s_x, s'_y))}{j} \tag{2.14}$$

Here $sim(s_x, s'_y)$ is the similarity between style s_x and style s'_y. Tag/Mood-based artist similarity can be obtained using the same approach.

In some applications, people may see the differences among these four different approaches due to the different scales of their results and the different ways they are associating with the terms in the hierarchies. In our system, however, we compared their results and did not see any significant differences among them after normalizing them into the same scale (0∼1). To further illustrate this, let us check the data distribution of the artist similarity values generated using these four different approaches.

We observe that there are almost no differences among the distributions of the artist similarity values calculated using 4 different approaches described above. Hence we use the average of all the 4 normalized quantified similarity values as the final artist similarity. We also observe that the tag-based and style-based artist similarity values are a little more diverse than the mood-based artist similarity values, therefore we use a heuristic proportion value to calculate the final combined artist similarity value. So we have:

$$c = 0.35 \times t + 0.35 \times s + 0.3 \times m \tag{2.15}$$

Here c, t, s, and m are respectively the combined artist similarity, the tag-based similarity, the style-based similarity, and the mood-based similarity. In the system, the similarity ranges between 0 and 1, where 0 is the most different and 1 is the most similar.

We are interested in how these user-assigned tags, *professionally assigned* mood and style terms are grouped together in describing artists. We believe that neither acoustic similarity nor T/S/M labels provide sufficient information to enable accurate similarity calculation. We are rather interested in how related the label-based similarity and the acoustics-based similarity are to each other. To explore more on this question, it would be ideal if we had acoustics data for all the 403 artists in the study, but the time and cost required for collecting the data would be prohibitive. So for the experimental study in the section, we chose to look at a limited number of artists. We present a case study on four famous artists (bands):

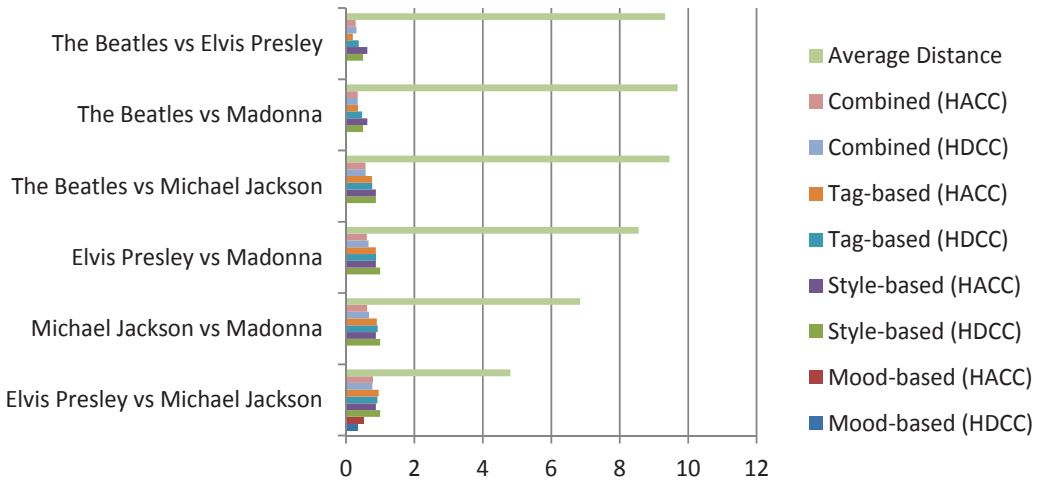


Figure 2.11: Results of the case study.

The Beatles, Elvis Presley, Madonna, and Michael Jackson to demonstrate the effectiveness of our framework. The quantified artist similarities among them are presented in Figure 2.11.

2.6.2 Music Feature Extraction

To show the performance of our HCC method in terms of its ability for computing the artist similarity, the artist similarity based on the music signals is also computed.

There has been a considerable amount of work in extracting descriptive features from music signals for music genre classification and artist identification. In our study, we use timbral features along with wavelet coefficient histograms. The feature set consists of the following three parts and 80 features in total, which can well reflect the moods and styles of the corresponding artists [FU01, TC02, LS01, LOL03].

Mel-Frequency Cepstral Coefficients (MFCC) is a feature set that is highly popular in speech processing. It is designed to capture short-term spectral-based features. The features are computed as follows: First, for each frame, the logarithm of the amplitude spectrum based on short-term Fourier transform is calculated, where the frequencies are divided into thirteen bins using the Mel-

frequency scaling. Next, this vector is decorrelated using discrete cosine transform. This is the MFCC vector.

Short-Term Fourier Transform Features (STFT) is a set of features related to timbral textures and is not captured using MFCC. It consists of the following five types: Spectral Centroid, Spectral Rolloff, Spectral Flux, Zero Crossings, and Low Energy. More detailed descriptions of STFT can be found in [TC02].

Daubechies Wavelet Coefficient Histograms (DWCH): Daubechies wavelet filters are a set of filters that are popular in image retrieval. The Daubechies Wavelet Coefficient Histograms, proposed in [LOL03], are features extracted in the following manner: First, the Daubechies-8 (db_8) filter with seven levels of decomposition (or seven subbands) is applied to 30 seconds of monaural audio signals. Then, the histogram of the wavelet coefficients is computed at each subband. Then the first three moments of a histogram, i.e., the average, the variance, and the skewness, are calculated from each subband. In addition, the subband energy, defined as the mean of the absolute value of the coefficients, is computed from each subband. More details of DWCH can be found in [LOL03, LO06].

2.6.3 Result Analysis

For each artist (band), we randomly pick 5 songs and conduct the following procedure. First, we exact the acoustic features of each song using the approach explained above. Then we calculate the pairwise Euclidean distances between the acoustic features that represent the songs of different artists (bands). Finally we calculate the average of all the pairwise distances as the average distance of the two artists. The results (average distance) are presented in Figure 2.11.

From the results, we observe that our quantified artist similarities match very closely the artist similarity based on the acoustic features of their music recordings. By checking the top two sets of bars in Figure 2.11, we can easily observe that the data variation trends from the top to the bottom, i.e, while the average distance

increases one by one, the combined similarity decreases almost constantly. In other words, the acoustic features of songs from the artists with higher similarity values (e.g., Elvis Presley versus Michael Jackson) are closer than those of songs from the artists with lower similarity values (e.g., The Beatles versus Madonna, and The Beatles versus Elvis Presley), while the acoustic features of songs from the artists with lower similarity values (e.g., Elvis Presley and The Beatles) are farther than those of songs from the artists with higher similarity values (e.g., Elvis Presley and Michael Jackson).

So far, HCC provides a reasonable alternative in addition to the standard way of computing artists similarity. It has potential to be extended to other social networks and help with quantifying the user similarity there.

2.7 Conclusion

In this part of dissertation, we systematically study the usage of hierarchical co-clustering methods for organizing different types of social network data. In particular, we experiment a previous hierarchical divisive co-clustering method and propose a novel hierarchical agglomerative co-clustering method. We utilize these two HCC methods to organize the social network data, so that the better and deeper understanding of the relationship between users and the user-generated content information (topic/tag/keyword) can be acquired. We perform experiments on real world music social network data sets and the experimental results show that HCC methods have the capability of generating good dendrograms with a global picture of users and the user-generated information as shown in Figure 2.2. In particular, our proposed HACC method outperforms the other competitors (results are shown in Figure 2.8). Furthermore, we observe that the HCC is able to incorporate instance-level constraints on users and/or user-generated content information to improve the clustering process (see Figure 2.9 and 2.10 for the comparison results). Last but not the least, a case study is conducted to show that

besides organizing the social network data, the HCC method can also be utilized for the quantification of the user similarity.

CHAPTER 3

A SUMMARIZATION FRAMEWORK OF TEXTUAL CONTENTS

Multi-document summarization is an important issue in the Information Retrieval community. It aims to distill the most important information from a set of documents to generate a compressed summary.

Recently, multi-document summarization is adopted in the social network domain since (1) many social network services are considered to be the so-called social media, and there are many text information generated by social network users in the form of blogs, tweets, reviews, comments, questions and answers, etc. (2) The audience of social media poses the requirement of quickly capture the ideas of news and hot trends, and an accurate and succinct summary is believed to be helpful in this case.

Audience in the social network tend to pose various different summarization requirements about the news and trends. (1) Some might only want the general summary of the textual contents in the social media; (2) some might be interested in the topic-specific summary; (3) some might care about the summary of the updates given a specific topic and an existing summary; (4) others might be interested in a comparison between two different summaries over the same topic. How does one fulfill so many requirements?

Given a set of documents as input, most of existing multi-document summarization approaches utilize different sentence selection techniques to extract a set of sentences from the document set as the summary. The submodularity hidden in the *term coverage* and the *textual-unit similarity* motivates us to incorporate this property into our solution to multi-document summarization tasks. In this chapter, we propose a new principled and versatile framework for different multi-document summarization tasks using submodular functions [NWF78] based on the term coverage and the textual-unit similarity which can be efficiently optimized through the improved greedy algorithm. We show that four known summarization

tasks, including generic, query-focused, update, and comparative summarization, can be modeled as different variations derived from the proposed framework. Experiments on benchmark summarization data sets (e.g., DUC04-06, TAC08, TDT2 corpora) are conducted to demonstrate the efficacy and effectiveness of our proposed framework for the general multi-document summarization tasks.

3.1 Overview

Multi-document summarization, as a fundamental and effective pattern for the document understanding and organization, enables better information services by creating concise and informative reports for a large collection of documents. It is useful for many real world applications. For example, Chen and Liu [CL09] aimed at tracking user-interested news events from a large pool of news. In this case, multi-document summarization can be applied to summarize those news events. A huge amount of available online textual documents in the field of biomedicine leads to great difficulties for building question answering, or information retrieval systems [DVA09]. Luckily, multi-document summarization can assist extracting the essential information from those documents and hereby benefit those systems.

Notice that the generated summary is either generic where the important information contained in input documents without any particular information needs is extracted, or query/topic-focused in which it is produced in response to a user query [JMK+00, Man01]. Recently, new summarization tasks such as the update summarization [DO08] and the comparative summarization [WZLG09] have also been proposed. The update summarization aims to generate short summaries of recent documents to capture new information different from earlier documents, and the comparative summarization focuses on summarizing the differences between comparable document groups.

The social-network audience poses different summarization requirements to understand the news and trends in Social Media. In this chapter, we propose

a new principled and versatile framework for **Multi-document Summarization** using the **Submodular Function** (MSSF) to help the social-network audience. Many known summarization tasks described above, including generic, query-focused, update, and comparative summarization, can be modeled as different variations derived from the proposed framework. The framework provides an elegant basis to establish connections between various summarization tasks while highlighting their differences.

In our summarization framework, the multi-document summarization problem is first mapped to the *budgeted maximum coverage problem* [KMN99] which often arises in circuit layout, job scheduling, facility location, and other areas. Then the submodularity underlying the *term coverage* and the *textual-unit similarity* is taken into consideration for the greedy summarization algorithm, and is shown to have the ability of addressing the multi-document summarization problem. We further take advantage of the submodularity to modify the general greedy algorithm and finally adopt this modified version to improve the efficiency of our framework for different multi-document summarization tasks. Our work is closely related to Lin et al. [LB10]. Different from their work which mainly resolves the generic summarization problem using the textual-unit (e.g., sentence) similarity, our work demonstrates advantages from three aspects:

1. proposes a new principled and versatile framework to address different summarization problems;
2. utilizes the improved greedy algorithm proposed by Minoux [Min78] which provides more efficiency than the general one as the backbone of the computation;
3. considers the term-coverage-based submodular function which shows the performance improvement over the textual-unit-similarity-based one.

3.2 Related Work

3.2.1 Generic Summarization

For generic summarization, a saliency score is usually assigned to each sentence, and then sentences are ranked according to the saliency score. Scores are usually calculated based on a combination of statistical and linguistic features. MEAD [RJST04], a well-known toolkit for document summarization, is an implementation of the centroid-based method in which sentence scores are computed based on sentence-level and inter-sentence features. In addition, there are some other approaches, including the probabilistic model [SJ04], non-negative matrix factorization based model [WLZD08] and graph-based model [ER04, WYX07b].

Lin et al. [LB10] propose attacking the generic multi-docum-ent summarization problem via submodular function. Our work shares the similar idea with theirs. However, their method only uses textual-unit-similarity (e.g., sentence-similarity) based submodular function, while ours also considers term-coverage based submodular functions which are more convincing under specific scenarios. Moreover, we also propose a principled and versatile framework which shows the capability to deal with many other summarization tasks besides the generic one. Last but not least, our method is more efficient due to the improved greedy algorithm.

3.2.2 Query-Focused Summarization

In query-focused summarization, the information of the given topic or query should be incorporated into summarizers and sentences suiting the user's declared information need should be extracted. Many methods for the generic summarization can be extended to incorporate the query information [SBC03, WLLH08]. Wan et al. [WYX07a] make full use of both the relationship among all the sentences in the documents and the relationship between the given query and the sentences by manifold ranking. Probability models have also been proposed with

different assumptions on the generation process of documents and queries [DM06, HV09, TYC09].

3.2.3 Update and Comparative Summarization

Update summarization was introduced in Document Understanding Conference (DUC) 2007 [Dan07] and was a main task of the summarization track in Text Analysis Conference (TAC) 2008 [DO08]. It is required to summarize a set of documents under the assumption that the reader has already read and summarized the first set of documents as the main summary. To produce the update summary, some strategies are required to avoid the redundant information which has already been covered by the main summary. One of the most frequently used methods for removing the redundancy is Maximal Marginal Relevance (MMR) [GMCK00]. Comparative document summarization is proposed by Wang et. al. [WZLG09] to summarize differences between comparable document groups. A sentence selection approach is proposed in [WZLG09] to accurately discriminate the documents in different groups modeled by the conditional entropy.

3.2.4 Submodularity

In general, Submodularity, a diminishing returns policy, shows that adding an element to a smaller set contributes more than adding it to a larger set, and is naturally to be used for efficiently finding out the optimal solution (in our case, the summarization). The formal definition of Submodularity is given as follows.

Let E be a finite set and f be a real valued nondecreasing function defined on the subsets of E that satisfies

$$f(S) + f(T) \leq f(S \cup T) + f(S \cap T), \tag{3.1}$$

where S and T are both subsets of E, such a function f is called **submodular** function [NWF78]. A key observation is that submodular functions are closed under nonnegative linear combinations [LKG+07].

Particularly, several works contribute to maximizing the submodular function. For example, [G$^+$84, NW81] attacked the general unit cost submodular function maximization problem. They showed that for the monotonic increasing submodular function the greedy algorithm could achieve an approximation factor $(1-1/e)$. Khuller et al. [KMN99] presented an algorithm that achieves an approximation factor $(1-1/e)$ for the budgeted submodular function maximization problem.

3.3 Algorithm Using Submodular Function

3.3.1 Why Submodularity?

The connection between the submodularity and the multi-document summarization cannot be easily identified. To clarify this, an alternative property of submodularity named as *decreasing marginal value* is given by:

$$f(T \cup \{\varsigma\}) - f(T) \leq f(S \cup \{\varsigma\}) - f(S), \tag{3.2}$$

where $S \subseteq T$, S and T are two subsets of E, and $\varsigma \in E \setminus T$. Intuitively, through this property, by adding one element to a larger set T, the value increment of f can never be larger than that by adding one element to a smaller set S. This intuitive diminishing property exists in different areas, *e.g.*, in the social network, adding one new friend cannot increase more social influence for a more social group than for a less social group [LKG$^+$07].

The *budgeted maximum coverage problem* is then described as: given a set of elements E where each element is associated with an influence and a cost defined over a domain of these elements and a budget B, the goal is to find out a subset of E which has the largest possible influence while the total cost does not exceed B. This problem is NP-hard [KMN99]. However, [KMN99] proposed a greedy algorithm which picks up the element that increases the largest possible influence within the cost limit each time and it guarantees the influence of the result subset is $(1-1/e)$-approximation. Submodularity resides in each "pick up" step.

Based on the submodular function and the budgeted maximum coverage problem, we can derive the answer to the question: why do we use submodularity for the multi-document summarization task? Let us delve into multi-document summarization task from two directions: the first one is the *term coverage*, and the second one is the *textual-unit similarity*.

Term Coverage

A pool of sentences is formed for the given document set. The problem is how to pick up the most representative sentences from that pool as the summary of this document set[1] within the budget. Suppose the budget is the number of terms, the action of adding one candidate sentence is associated with its *summarization quality increase* (i.e., the overall quality increase incurred by the terms in this candidate sentence while not in the already picked sentences.) and *cost.* (i.e., the number of terms in this candidate sentence.) The quality of the current generated summary S over the document set is hereby defined as

$$f(S) = \#(\{t|t\ is\ term\ of\ S\}), \tag{3.3}$$

which denotes the cardinality of the term set of S. Accordingly, the quality increase incurred by adding a candidate sentence can be defined by

$$I(\varsigma) = \#(\{t_1|t_1\ is\ term\ of\ \varsigma\} \setminus \{t_2|t_2\ is\ term\ of\ S\}), \tag{3.4}$$

where ς is the candidate sentence.

It does make sense that the function f holds the submodular property since the quality increase given by a candidate sentence based on a larger set of already picked sentences is smaller than that based on a smaller set. One common practice in defining f is to assign the weight (we treat the term frequency as the weight in this chapter) to each term in the document set. Then the definition of $f(S)$ is

[1]Here, the number of the extracted sentences or the number of words inside these sentences is fixed. We treat it as the budget B.

given by the following equation:

$$f(S) = \sum_{t \in S} w_t, \tag{3.5}$$

where w_t is the weight of term t.

Accordingly, the definition of the quality increase incurred by adding a new sentence ς to the current generated summary S is

$$I(\varsigma) = \sum_{t \in \varsigma \setminus S} w_t, \tag{3.6}$$

Intuitively, the candidate sentence which provides more quality increase should be picked as the new sentence to form the summary and the length of the final summary is fixed. Hence, we can treat multi-document summarization as a budgeted maximum coverage problem.

Textual-Unit Similarity

If the budget B is the number of terms in the summary, the cost of each candidate sentence is the number of terms within it. A high quality summary should be two-fold: 1) maximizes the information coverage of the given document set; 2) minimizes the redundancy. One of the most popular methods for serving these two purposes is Maximal Marginal Relevance (MMR) [GMCK00] which aims to reduce the redundancy and maintain query relevance in retrieved documents at the same time. Hence, a MMR-similar definition for the quality of the current generated summary is given by

$$f(S) = \sum_{s_i \in E \setminus S} \sum_{s_j \in S} sim(s_i, s_j) \quad - \sum_{s_i, s_j \in S, s_i \neq s_j} sim(s_i, s_j), \tag{3.7}$$

where E is the whole sentence set, $sim(s_i, s_j)$ is the weight between the textual units s_i and s_j (the typical textual unit is sentence). Note that the first component of Eq.(3.7) is for the information coverage and the second component is for the redundancy removal, these two terms carry the same weight. Both information

coverage and redundancy terms of $f(S)$ are submodular, thus $f(S)$ is also submodular, since the linear combination of submodular functions is closed. Suppose ς is the candidate sentence, the quality increase is therefore represented as follows:

$$I(\varsigma) = f(S \cup \{\varsigma\}) - f(S). \tag{3.8}$$

The goal is to generate a summary which provides the largest possible quality within the budget. Hence, the mapping from the multi-document summarization problem to the budgeted maximum coverage problem is straightforward.

According to the above analysis, the multi-document summarization problem can be modeled as a budgeted maximum coverage problem in two different levels – the term coverage and the textual-unit similarity. The general greedy algorithm for the multi-document summarization is presented in Section 3.3.2.

3.3.2 Algorithm for Summarization

The main idea of the greedy algorithm for the multi-document summarization problem is simple: sequentially pick up the sentence which provides the largest quality increase based on the sentences in the current summary until the budget is reached.

As we discussed in Section 3.3.1, there are two ways of defining the specific submodular function for the summarization. The first one is from the term coverage perspective, and the second one is from the textual-unit similarity perspective. Given a document set D, a budget B and the indication of two submodular function types, the greedy algorithm utilizes the appropriate submodular function to generate a summary for D within B. The procedure is shown in Algorithm 3.

Algorithm Details

The core components in Algorithm 3 are "Summ-Term-Coverage", and "Summ-UnitSimilarity". Most details of these two core components are identical in Algo-

Algorithm 3 The greedy algorithm for summarization

Input DocumentSet D, budget B,
SubmodularFunctionType T
if $T =$ "$Term\ Coverage$" **then**
Summary = Summ-TermCoverage(D, B)
end if
if $T =$ "$Textual - Unit\ Similarity$" **then**
Summary = Summ-UnitSimilarity(D, B)
end if
Output Summary

rithm 4 except the definition of the submodular function f as well as the quality increase incurred by adding a new sentence.

Algorithm 4 The core component of the greedy algorithm for summarization

Extract sentence set E from document set D
Initial remaining sentence set R as E
Initial summary S as $\emptyset$
Initial cost C as 0

while Size(R)>0 **do**
$\varsigma \leftarrow$ The sentence which has $arg\ max_{e \in R} \frac{I(e)}{(length(e))^p}$
if $(C \leftarrow C + length(\varsigma)) < B$
$S \leftarrow S \cup \{\varsigma\}$
$R \leftarrow R \setminus \{\varsigma\}$
else
Stop
end if
end while
Return S

The definition of f for "Summ-TermCoverage" is given in Eq.(3.5). In this context, whenever the algorithm picks up a new sentence into the summary, it would somehow strive to choose the longer sentence in the remaining sentence set, since the longer one has more possibility to cover more important terms and provide more quality increase. To avoid the summary containing only long sentences of the document set, we include the length of sentence as denominator of the quality increase to weaken such effect. f for "Summ-UnitSimilarity" is the one defined in Eq.(3.7). Practically, we treat each sentence as the basic textual unit, and represent sentences as term vectors, each entry of which is the weight of

Summarization Type	***Submodular Function***
Generic Summarization	$f(S) = \sum_{t \in S} w_t$ $f(S) = \sum_{s_i \in D \setminus S} \sum_{s_j \in S} sim(s_i, s_j)$ $- \sum_{s_i, s_j \in S, s_i \neq s_j} sim(s_i, s_j)$
Query-focused Summarization	$f(S, q) = f_G + \sum_{s_i \in S} sim(q, s_i)$
Update Summarization	$f(q, S_1, S_2) = f_G + \sum_{s_i \in S_2} sim(q, s_i)$ $- \sum_{s_i \in S_2} \sum_{s_j \in S_1} sim(s_i, s_j)$
Comparative Summarization	$f(S) = f_G$ $- \sum_{s_i \in S} \sum_{s_j \in OtherGroups} sim(s_i, s_j)$

Table 3.1: A quick summary of the submodular functions for different summarization tasks.

term frequency-inverse sentence frequency (TF-ISF) [JMK+00]. Then the weight between two sentences is the pairwise cosine similarity. Similarly, we include the length of sentence as the denominator of the quality increase to avoid the bias.

Note that, in the case when the scaling factor of sentence length $p = 1$ and f is a normalized monotonic submodular function, it was proved by [KMN99] that Algorithm 4 achieves a bounded approximation ratio $(1 - 1/e^{\frac{1}{2}})$; in other cases when the number of the sentences in the final summary S is $|S|$, $0 \leq p < 1$ and f is a normalized monotonic submodular function, Lin et al. [LB10] proved that Algorithm 4 guarantees a bounded approximation ratio $(1 - \prod_{n=1}^{|S|} (1 - (c_n/B)^p))$. On one hand, Eq.(3.5) is a normalized monotonic submodular function, therefore, the above theoretical results holds; on the other hand, Lin et al. [LB10] proved that Algorithm 4 could still solve the summarization problem near-optimally with a high probability even though Eq.(3.7) is not guaranteed monotonic.

Improvements on Algorithm

As we can see, each time the greedy algorithm picks up a new sentence, it has to recompute the quality increases considering each of the remaining sentences as the candidate based on the current summary. Suppose the given document set contains a huge number of sentences, the running time would be unacceptable.

Hence, in order to apply this method to real world applications, we are wondering if the running time of this algorithm could be reduced.

Inspired by the work of Minoux [Min78], we further utilize the sumodularity to make modifications to the process of picking up new sentences. This general idea is: once the top sentence in the remaining sentence set R which holds the largest value of $\frac{I(e)}{(length(e))^p}$ based on the current summary, its following sentences can never surmount it, since as the summary enlarges, the value of $\frac{I(e)}{(length(e))^p}$ is getting smaller and smaller. In such case, there is no need to recompute $\frac{I(e)}{(length(e))^p}$ of all remaining sentences as in Algorithm 4, so that the running time is greatly reduced.

From Algorithm 5, one can find out the details of the changes.

Algorithm 5 The core component of the improved greedy algorithm

Extract sentence set E from document set D
Initial summary S as $\emptyset$
Initial remaining sentence set R as E
Assign $v_e = \frac{I(e)}{(length(e))^p}$ to each sentence e of R
Initial cost C as 0

while Size(R) > 0 **do**
  while true **do**
    t=Top(R)
    $R \leftarrow$ sort R based on v_e of each sentence e
    t'=Top(R)
    if $t \neq t'$
      Save $v'_t = \frac{I(t')}{(length(t'))^q}$ for t'
    else
      Stop the while
    end if
  end while
  if $(C \leftarrow C + length(t')) < B$
    $S \leftarrow S \cup \{t'\}$
    $R \leftarrow R \setminus \{t'\}$
  else
    Stop
  end if
  $t'' =$ Top(R)
  Save $v''_t = \frac{I(t'')}{(length(t''))^q}$ for t''
end while
Return S

3.4 The Summarization Framework

Our proposed submodularity-based framework can be modeled to different multi-document summarization tasks, including generic, query-focused, update and comparative summarization. In this section, we formulate each summarization task by defining different submodular functions.

For the generic summarization, we present the submodular function from the two aforementioned aspects: the term coverage and the textual-unit similarity. Table 3.1 summarizes the submodular functions for different summarization tasks, and Table 3.2 presents the notations. The general procedure of methods for different summarization tasks is described in Algorithm 3 and Algorithm 5, while the only difference resides in the submodular functions.

Notation	***Meaning***
D	Document Set
S	Summary
S_1	Summary for D_1
S_2	Summary for D_2
S'	Existing Summary
w_t	Weight of term t
s_i, s_j	Textual unit
sim	Similarity
q	Given query
f_G	General information coverage

Table 3.2: Notations.

3.4.1 Generic Summarization

Given a set of documents, the generic summarization is the task of extracting a set of sentences which can cover the general ideas of the document set. If there is no length limit to the summary, all the sentences in the whole document set would be the final summary since they cover all the content of the documents. However, such summary results in great difficulty of reading and capturing the general ideas for users; contrarily, it would be better to set a summary length for

the summarization task. As discussed in Section 3.3.2, given the length limit to the summary, the generic summarization problem can be resolved by using the submodular function.

The submodular function for the generic summarization is defined as Eq.(3.5) for the term frequency or Eq.(3.7) for the textual-unit similarity. Notice that Eq.(3.5) considers all terms no matter how many times they appear in the document set. In reality, it is possible that the result will involve terms with lower frequency and with no remarkable contribution to the summary. Therefore, we set the threshold λ in the experiment to filter such terms. In other words, if the frequency of a term is less than λ, it will be discarded.

3.4.2 Query-Focused Summarization

The query-focused summarization is to generate a short summary based on a given document set and a given query. The generated summary reflects the condensed information related to the given query under the length budget. Different from the generic summarization that generates summaries presenting the general ideas of the document set, the query-focused summarization provides the summary that can satisfy special requirement for users.

Given a document set and a query q, we define the quality function as

$$f(S, q) = f_G + \sum_{s_i \in S} sim(q, s_i), \tag{3.9}$$

where the first term represents the general information coverage which could be replaced by Eq.(3.5) or Eq.(3.7), the second term represents the query-focused information coverage. Clearly, this function is a submodular function, since both parts in Eq.(3.9) are submodular, and the linear combination of submodular functions is closed.

3.4.3 Update Summarization

The update Summarization is a form of the multi-document summarization in which we generate a summary of a new document set based on the assumption that the user has already read a given document set. Generally, this summarization task is based on the following scenario: A user is interested in a particular news topic and wants to track its related news as it evolves over time, so he/she subscribes to a news feed that sends his/her relevant articles as they are submitted from various news services. However, either there are so many news that he/she cannot keep up with, or he/she has to leave for a while and then wants to catch up. Whenever he/she checks out news of his/her interested topic, it bothers him/her that most articles keep repeating the same information; he/she would like to read summaries that only talk about what's new or different about this topic.[2] We formulate such problem as follows:

Given a query q (represents the user's interested topic) and two sets of documents D_1 (already read articles) and D_2 (new articles), the update summarization aims to generate a summary of D_2 related to the query q, given D_1. First of all, the summary of D_1, referred as to S_1, can be generated. Then, the update summary of D_2 related to q, referred as to S_2 is generated. The main idea of S_2 should be different from the main idea of S_1. Also, S_2 should cover all the aspects of the document set D_2 as many as possible.

Based on this formal definition, we formulate the submodular function for the update summarization as

$$f(q, S_1, S_2) = f_G + \sum_{s_i \in S_2} sim(q, s_i) \quad - \sum_{s_i \in S_2} \sum_{s_j \in S_1} sim(s_i, s_j), \tag{3.10}$$

where S_1 is the existing summary and S_2 is the updated summary. The first term in Eq.(3.10) denotes the general information coverage for new coming document

[2]http://www.nist.gov/tac/2009/Summarization/update.summ.09.guidelines.html

set, the second term denotes the query-focused information coverage and the last terms denotes the redundancy given S_1.

Since each term is a submodular function, the property of the submodularity holds for the linear combination of these terms. Similar to Eq.(3.7), Eq.(3.10) is not monotonic everywhere, but there is a high probability that a near-optimal solution can be generated.

3.4.4 Comparative Summarization

Given a collection of document groups, the comparative summarization is to generate a short summary delivering the differences of these documents by extracting the most discriminative sentences in each document group. The traditional document summarization aims to cover the majority of the information among document collections, while the comparative summarization is to find differences. We formulate the comparative summarization as follows:

Given N groups of documents $G_1, G_2, \cdots, G_N$, the comparative summarization aims to generate summaries $S_1, S_2, \cdots, S_N$ such that the summaries can represent topics of corresponding groups whereas they are different from each other on the theme level.

We extend our greedy algorithm for the comparative summarization to generate the discriminant summary for each group of documents. The submodular function for the comparative summarization is defined as

$$f(S) = f_G - \sum_{s_i \in S} \sum_{s_j \in OtherGroups} sim(s_i, s_j), \tag{3.11}$$

where S is the summary of the current group. The first term represents the general information coverage of current group, while the second term represents the redundancy based on the other groups. Clearly, the linear combination of these terms holds the submodularity property. As before, without the monotonic property, Eq.(3.11) has a high probability to generate a near-optimal solution.

	DUC04	***DUC05***	***DUC06***	***TAC08 A***	***TAC08 B***
Type of Summarization	Generic	Query-focused	Query-focused	Query-focused	Update
#topics	50	50	50	48	48
#documents per topic	10	25-50	25	10	10
Summary length	665 bytes	250 words	250 words	100 words	100 words

Table 3.3: Brief description of the data sets.

3.5 Experiments

Since there is not a benchmark data set for the social network textual contents, this evaluation uses the benchmark data sets coming from the multi-document summarization community.

We have conducted experiments on the four summarization tasks and our proposed method based on the submodular function has outperformed many existing approaches. For the generic summarization, the DUC04 data set is applied. For the query-focused summarization, the DUC05 and the DUC06 data sets are adopted as the experiment data. As for the update summarization task, the experiments are performed on the TAC08 data set. For the comparative summarization, we use the TDT2 corpora to compare the summary generated by different comparative summarization methods. Note that we treat the sentence as the basic textual unit for all experiments those need consider textual-unit similarity.

All the tasks, except the comparative summarization, are evaluated by Recall-Oriented Understudy for Gisting Evaluation (ROUGE) – an evaluation toolkit for document summarization [Lin04] which automatically determines the quality of a summary by comparing it with the human generated summaries through counting the number of their overlapping textual units (e.g., n-gram, word sequences, and etc.). In particular, F-measure scores of ROUGE-2 and ROUGE-SU4 are presented for our experiment. For the comparative summarization, we provide two other approaches for the purpose of comparison. The detailed experimental results are described in the following.

3.5.1 Generic Summarization

For the Generic summarization, we use DUC04 as the experimental data. We evaluate our method (denoted as MSSF) on the generic summarization from two aspects: the term coverage and the textual-unit similarity (in our experiment setup, the sentence is the basic textual-unit).

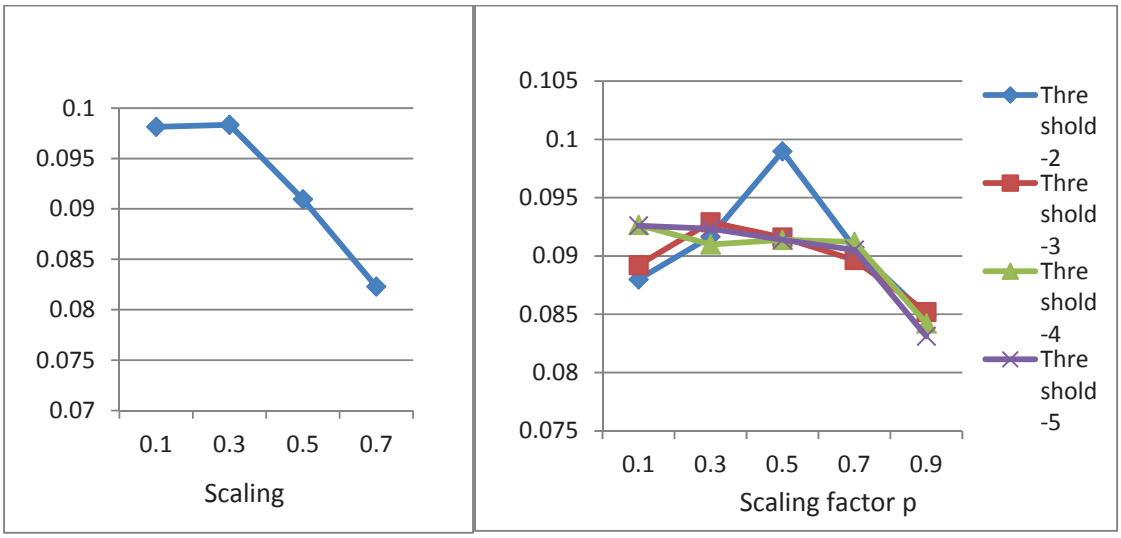


Figure 3.1: Left: ROUGE-2 for MSSF(Sentence Similarity) using scaling factor 0.1-0.7; Right: ROUGE-2 on threshold ranging from 2-5 for MSSF(Term Coverage) using scaling factor 0.1-0.9.

	ROUGE-2	***ROUGE-SU4***
DUC Best	0.09216	0.13233
Random	0.06377	0.11779
MMR	0.09144	0.13287
LexPageRank	0.08572	0.13097
Centroid	0.07379	0.12511
LSA	0.06538	0.11946
NMF	0.07261	0.12918
MSSF(Term Coverage)	*0.09897*	*0.13951*
MSSF(Textual-Unit Similarity)	*0.09834*	*0.13901*

Table 3.4: Results on generic summarization.

	DUC05		***DUC06***	
	ROUGE-2	*ROUGE-SU4*	*ROUGE-2*	*ROUGE-SU4*
Average-Human	0.10236	0.16221	0.11249	0.1706
DUC Average	0.06024	0.11488	0.07543	0.13206
Random	0.04143	0.09066	0.04892	0.10083
LSA	0.04079	0.09352	0.05022	0.10226
SNMF	0.06043	0.12298	0.08549	0.13981
Qs-MRF	0.0779	0.1366	0.08917	0.14329
Wiki	0.07074	0.13002	0.08091	0.14022
MSSF	0.0731	0.12718	0.09193	0.14611

Table 3.5: Results on query-focused summarization.

We observe through the experiment that the summary result generated by our method is the best when the threshold $\lambda = 2$. Consequently, we set λ as 2 when performing comparative experiments with other existing methods. we also conduct experiments to evaluate the sensitivity of the scaling factor p on MSSF (Textual-Unit Similarity) and on MSSF (Term Coverage) using different thresholds. From Figure 3.1 , we have two observations: (1)Different scaling factors do have different impact on the result under consistent experiment setting (here, consistent setting could mean MSSF (Textual-Unit Similarity) or the same threshold on MSSF (Term Coverage)); (2)Under different experiment settings, the best results are not always given by the same scaling factor, i.e., when performing MSSF (Term Coverage) using the threshold 2, the best summarization is given by the scaling factor 0.5, while performing MSSF (Term Coverage) using the threshold 3, the scaling factor 0.3 gives the best result.

After clarifying the impact of the scaling factor, we set it as 0.5 for MSSF (Term Coverage) and 0.3 (It shows the best result in Figure 3.1 when p is set as 0.3) for MSSF (Sentence Similarity). We implement the following widely used or recent published methods for generic summarization as the baseline systems to compare with our proposed method MSSF: (1) Random: the method randomly selects specific sentences as the summary; (2) Maximum Marginal Relevance (MMR): the method is similar with MSSF (Textual-Unit Similarity) as we mentioned in Section 3.3.1. It greedily selects the sentence which maximizes the relevance with the given document set while minimizes the redundancy with the sentences that have already been selected; (3) LexPageRank: the method first constructs a sentence connectivity graph based on the cosine similarity and then selects important sentences based on the concept of eigenvector centrality [ER04]; (4) Centroid: the method extracts sentences based on the centroid value, the positional value and the first sentence overlap; (5) Latent Semantic Analysis (LSA): the method identifies semantically important sentences by conducting latent semantic analysis;

(6) Non-negative Matrix Factorization (NMF): the method performs NMF on the sentence-term matrix and select the high ranked sentences.

From the results showed in Table 3.4, our method MSSF clearly outperforms the other rivals and is even better than the DUC04 best team work. Note that MSSF (Term Coverage) is slightly better than MSSF (Textual-Unit Similarity) which has the similar submodular function as the work of Lin et al. [LB10]. Since one sentence of the given document set should be covered by at least one sentence in the summary, not by all summary sentences, sometimes there may exist bias in the first term of submodular function Eq.(3.7). In a word, MSSF (Term Coverage) is more reasonable.

3.5.2 Query-Focused Summarization

	TAC08 A		***TAC08 B***	
	ROUGE-2	*ROUGE-SU4*	*ROUGE-2*	*ROUGE-SU4*
TAC Best	0.1114	0.14298	0.10108	0.13669
TAC Median	0.08123	0.11975	0.06927	0.11046
MSSF	0.08327	0.12109	0.09451	0.13180

Table 3.6: Results on update summarization.

Main tasks of DUC05 and DUC06 are both the query-focused summarization, and therefore we conduct experiments on these two data sets. In addition to baseline systems, we also compared our system with some widely used and recently published systems: (1) SNMF [WLZD08]: calculates sentence-sentence similarities by the sentence level semantic analysis, clusters the sentences via the symmetric non-negative matrix factorization, and extracts the sentences based on the clustering result; (2) Qs-MRF [WLLH08]: extends the mutual reinforcement principle between the sentence and the term to the document-sentence-term mutual reinforcement chain, and uses the query-sensitive similarity to measure the affinity between the pair of texts; (3) Wiki [Nas08]: uses Wikipedia as the external knowl-

edge to expand the query and builds the connection between the query and the sentences in documents.

The empirical result are reported in Table 3.5. The results show that on DUC05, our method outperforms the the other systems except Qs-MRF and Wiki; on DUC06, our method achieves the best result. This is due to the novel adoption of the submodular function. Note that our method is simpler than the other systems because of the greedy heuristic.

3.5.3 Update Summarization

Note that the data set for the update summarization, (i.e. the main task of TAC08 summarization track), is composed of 48 topics and 20 news wire articles for each topic. The 20 articles are grouped into two groups. The brief description can be found in Table 3.3. The update summarization task requires to produce two summaries, involving the initial summary (TAC08 A), which is the standard query-focused summarization, and the update summary (TAC08 B) under the assumption that the reader has already read the first 10 documents.

Table 3.6 shows the comparative experimental results on the update summarization. In Table 3.6, "TAC Best" and "TAC Median" represent the best and median results from the participants of the TAC08 summarization track in the two tasks respectively according to the TAC08 report [DO08]. As seen from the results, the ROUGE scores of our methods are higher than the median results. The good results of the best team typically come from the fact that they utilize advanced natural language processing (NLP) techniques to resolve pronouns and other anaphoric expressions. Although we can spend more efforts on the preprocessing or the language processing step, our goal here is to demonstrate the effectiveness of formalizing the update summarization problem using the submodular function and hence we do not utilize advanced NLP techniques for preprocessing. Experimental results demonstrate that our simple update summarization method

based on the submodular function can lead to the competitive performance for the update summarization.

Topic	***Description***
1	Iraq Issues
2	Asia's economic crisis
3	Lewinsky scandal
4	Nagano Olympic Games
5	Nuclear Issues in Indian and pakistan
6	Jakarta Riot

Table 3.7: TDT2 corpora topic description.

3.5.4 Comparative Summarization

For the comparative summarization, we use the top six largest clusters of documents from the TDT2 corpora to compare the summary generated by different comparative summarization methods. The topics of the six document clusters are described as in Table 3.7.

From each of the topics, 30 documents are extracted randomly to produce a one-sentence summary. For the comparison purpose, we select the sentence that is the most similar to other sentences in the document group as the baseline, denoted as "MS". We also implement the methods proposed by [WZLG09]. Table 3.8 shows the summaries generated by MS, the discriminative sentence selection (DSS) [WZLG09] and our method MSSF. As we can see, DSS can extract discriminative sentences for all the topics except topic 4 and topic 6. Note that the sentence extracted by DSS for topic 4 may be discriminative from other topics, but it is deviated from the topic Nagano Olympic Games. The MS method can extract general sentences related to the corresponding topics to some extent. However, some sentences extracted by MS only contains the keywords of the related topics, but not the essence of the topic (i.e., the summary of topic 3). Comparatively, our MSSF method can extract discriminative sentences for all topics with the essential

Topic	*MS*	*DSS*	*MSSF*
1	··· U.S. Secretary of State Madeleine Albright arrives to consult on the stand-off between the **United Nations and Iraq**.	**the U.S. envoy to the United Nations**, Bill Richardson, ··· play down China's refusal to support threats of **military force against Iraq**	The arrival of U.S. Secretary of State Madeleine Albright could be an early test of **the accord iraq signed ten days ago with U.N. secretary-general Kofi Annan**.
2	**Thailand's currency**, the baht, **dropped through a key psychological level** of ··· amid a regional sell-off sparked by escalating **social unrest in Indonesia**.	Earlier, driven largely by **the declining yen**, **South Korea's stock market fell** by ···, while the **Nikkei 225 benchmark index dipped** below 15,000 in the morning ···	Prueher addressed the army seminar in Manila and told delegates that **Asia's financial troubles** have affected **the United States' joint military activities with its Asian allies**.
3	··· attorneys representing **President Clinton and Monica Lewinsky.**	The following night **Isikoff** ···, where he directly followed the recitation of the top-10 list: "Top 10 **White House Jobs That Sound Dirty.**"	In Washington, Ken Starr's grand jury continued its investigation of the **Monica Lewinsky matter**.
4	Eight women and six men were named Saturday night as the first **U.S. Olympic Snowboard Team** as their sport gets set to make its debut in **Nagano, Japan**.	*this tunnel is finland's cross country version of tokyo's alpine ski dome, and olympic skiers flock from russia, ···, france and austria this past summer to work out the kinks* ···	Seizinger, **the German all-round skier** who has been **Street's fiercest rival** in recent years, did win **the downhill title**.
5	**U.S. officials** have announced **sanctions** Washington will impose on **India and Pakistan** for conducting **nuclear tests**.	The **sanctions** would stop all foreign aid except for humanitarian purposes, **ban military sales to India** ···	**Weapons experts** say **Pakistan** has long thought to have had all the components necessary to **build a nuclear device**.
6	··· remain in force around **Jakarta**, and at the Parliament building where **thousands of students staged a sit-in** Tuesday ···.	President Suharto has given much to his country over the past 30 years, raising Indonesia's standing in the world ···	··· to **press their demand** for what they feel should be true **political reform**, that is, **election of a totally new government** ···.

Table 3.8: A case study on comparative document summarization. Some unimportant words are skipped due to the space limit. The bold font is used to annotate the phrases that are highly related with the topics, and italic font is used to highlight the sentences that are not proper to be used in the summary.

idea. For example, the summary of topic 6 clearly explains the reason why the Jakarta Riot happened.

3.5.5 Improved Algorithm

To evaluate the efficiency of the improved greedy algorithm used in our summarization framework, we compare the general greedy algorithm with this new algorithm for generic, query-focused and update summarization tasks on DUC04, DUC05 and TAC08 accordingly. Notice that the summaries generated by the general greedy algorithm and the improved one are the same. For each summarization task, we perform each of the two algorithms for ten times, and compute the average running time for each of them. The comparison results are shown in Figure 3.2. We observe that the improved algorithm is shown to be more efficient on all the tasks. The running time comparison demonstrates the efficiency of our proposed summarization framework.

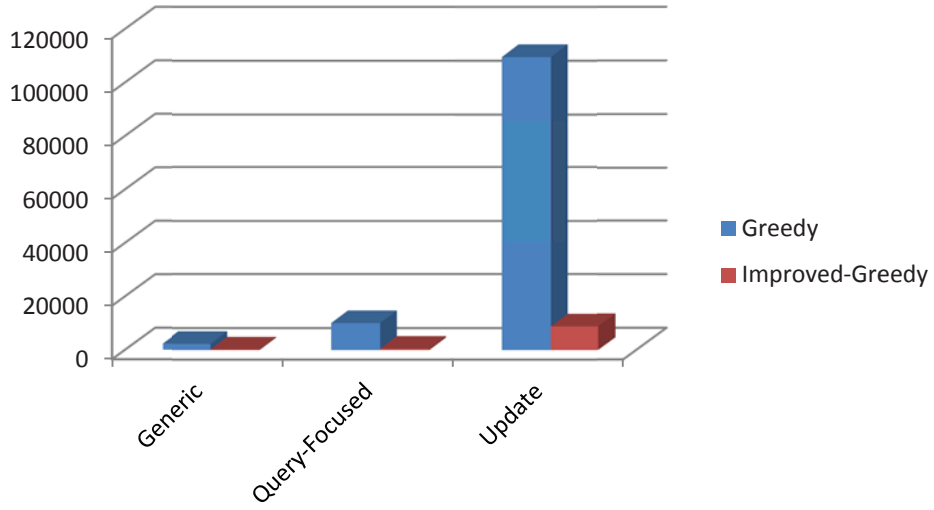


Figure 3.2: Average running time (in milliseconds) of two algorithms on three summarization tasks.

3.6 Conclusion

In this chapter, we present a new principled and versatile summarization framework – MSSF for **Multi-document Summarization** using the **Submodular**

Function. This framework can deal with different summarization tasks in the social network, including generic, query-focused, updated, comparative summarization. The empirical results show that this framework outperforms the other rivals in the generic summarization and is competitive in other summarization tasks. The ability to address these summarization problems benefits from various submodular functions for corresponding summarization tasks. Our proposed framework is shown to be more efficient because of the proposed improved summarization algorithm.

CHAPTER 4

SUMMARIZATION FOR TIME-SENSITIVE OSN CONTENTS

The Time-Sensitive OSN content is the root cause for audience to select Social Media for obtaining the latest news and updates from all over the world. It has the following characteristics in terms of its adjective — Time-Sensitive: (1) an event regarding a particular topic evolves quite fast in the Social Media. It is usually difficult for an individual to go over every news/updates/messages related to this event. (2) A huge number of events happen at the same time, even though not every event deserves the attention. (3) One event about one topic might be associated with many messages/news, which carry different sentiments at the same time.

Three research problems arise due to the characteristics of the time-sensitive OSN content. (1) Given an event about a topic, how does one capture the big picture of its causes and effects without referring to every news/updates/messages of this event? (2) How does one help audience identify their interesting events as the news/updates/messages are keeping coming? (3) How does one summarize the coming news/updates/messages based on different topics and different sentiments?

To address these three research problems, we conduct research along three tracks, respectively. Specifically, we (1) propose to sketch a real-time storyline of the event by a multi-document summarization solution, given an on-going event; (2) explore the application of an anomaly detection method over the time-sensitive OSN data for identifying interesting events; (3) adopt the traditional classification methods for attacking a "multi-task multi-label" (in this dissertation, multi-task means topic classification and sentiment classification tasks, while multi-label indicates that there are more than two pre-defined classification labels for each of the task.) classification problem. These three threads of research indicate different ways to summarize the time-sensitive contents.

It is worth noticing that Microblog is the most well-known example of time-sensitive OSN contents, and Twitter is currently the most famous Microblog service with the textual message — Tweet as its microblog. Therefore, Tweet, Microblog, or time-sensitive OSN contents are indicating the same textual social-network message in this chapter.

4.1 Overview

4.1.1 Storyline Generation

Microblogging service has rapidly increased its popularity in recent years. People are attracted to microblogging sites, such as Twitter, for instant first-hand reports on real-life events. In the meantime, instead of using web search engines, users are more willing to propose event queries on Twitter to obtain information about an ongoing event [TRM11]. Systems that deliver realtime event notification on Twitter are also available [SOM10].

It would be helpful for industry, academia, and end-users, if a skeleton of an event by request is automatically generated from the huge volume of tweets. We refer this problem as *Generating Event Storyline from Microblogs (GESM)*. For example, Figure 4.1 presents the storyline based on an event query of "Egypt Revolution". The vertical location of each frame indicates the time-stamp of the corresponding phase. The hierarchical structure depicts how major progress happens in adjacent phases. The branches partition simultaneously happened tweets into different semantic groups. Auto-generated storylines facilitates easy navigation in microblogoshpere and also supports a wide range of mining systems on collective intelligence.

GESM is a challenging problem. There exist studies in generating storylines from news articles [LL08, WZLD09, KMS04]. However, few works are proposed for the massive social network microblog data [1]. In order to provide a reasonable

[1]Twitter is currently the most famous microblog service in the world. Tweet is the

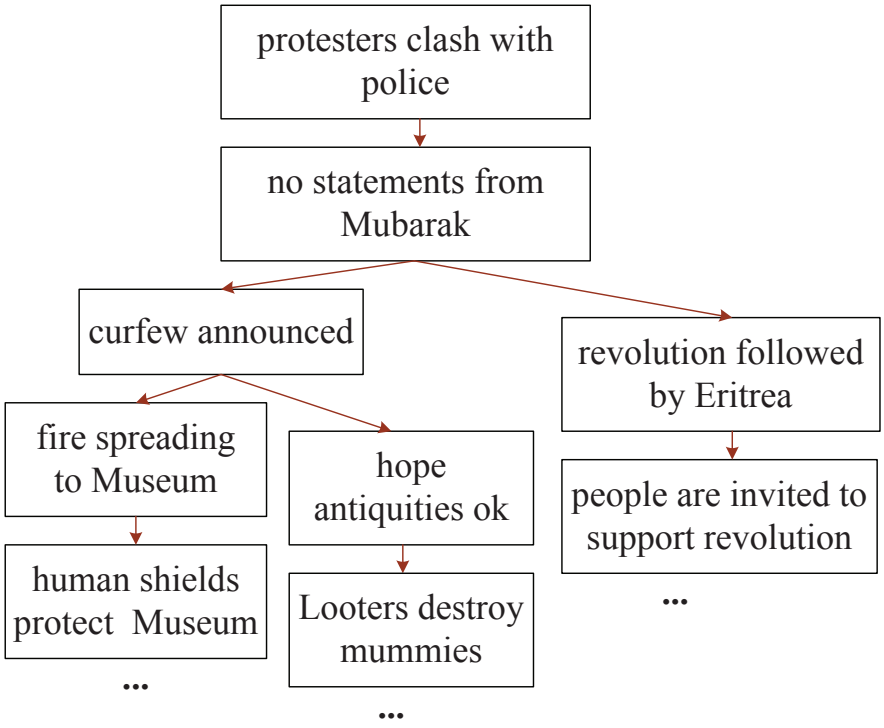


Figure 4.1: A sample storyline for event query – Egypt Revolution.

storyline to the social network audience in response to his/her query, a two-level framework is necessary: in the low level, finding all relevant microblogs through the time-line of the event by a retrieve model; and in the high level, summarizing relevant microblogs and the latent structure to produce a storyline. In this dissertation, we will not cover the technique details of the retrieve model for identifying relevant microblogs (Tweets) given a query (one can refer to the paper [LLL+12b] for all details.). Rather, we will focus on summarizing relevant microblogs to produce storylines.

Challenges of microblog storyline generation arise from the following aspects. The social nature of tweets increases the difficulty of integrating semantic similarity with chronological order in generating a storyline. Information sharing in microblog sphere yields numerous duplicate tweets and direct and undirect re-tweets. Duplicate tweets and re-tweets are created after the right time point, and they

name for the microblog generated on Twitter. Thus, in the following sections, the terms — tweet and microblog are used interchangeably.

will trigger confusion in partitioning the event time-line. Thus a naive method, which employs traditional text summarization strategy in each time segment, is not applicable.

In this work, we focus on resolving the above challenges. Major contributions of this work are: (1) A novel problem of generating event storylines from microblogs is proposed. (2) The problem of storyline generation on the retrieved microblogs is formulated as a graph-based optimization problem and is solved by approximation algorithms of minimum-weight dominating set and directed Steiner tree. The generated storylines ensure both temporal continuity and content coherence.

4.1.2 Event Detection

Multi-document summarization, such as the aforementioned storyline generation can be categorized as a classical way to summarize the OSN contents. The problem is, how does one discovery the new interesting events in the social media if he/she does not care about the summarized description of the event?

Event is something that occurs in a certain place at a certain time. Event detection describes the task which automatically detects events based on the given data (in our case, the data usually indicates the tweets). This research topic has been attracting attention from different areas: surveillance, scientific discovery, fault detection, anomaly detection, etc. [KRS].

In this work, to fulfill the social-media audience's requirement of receiving his/her interesting events in a timely fashion, we propose collecting topic specific tweets for the event detection because of the large number of active Twitter users.

Considering the nature of the tweet, traditional event detection methods may not be the best fit. Most of the existing event detection techniques are inheriting the idea of the clustering algorithm [PA+98, APL98, YPC98], either of the batch version (assuming all tweets are given, and then performing clustering for only one time over all the given tweets) or of the streaming version (assuming tweets are

keeping coming, and processing one tweet at one time). The drawbacks of these methods are obvious.

First, clustering techniques usually require users to indicate a specific cluster number beforehand. However, it is widely accepted that the cluster number is difficult to decide. Although some incremental clustering methods do not define a specific cluster number at the very beginning, they still have to define some "similarity/distance threshold" to differentiate the data point which is far from any existing clusters, create a new cluster for it, and then put it into the new cluster.

Second, given tweet clusters by clustering algorithm, we have no idea on interpreting these results. The clustering algorithm groups similar data points together, while separates non-similar data points. In this case, does a cluster mean a specific event which is described by similar tweets within it? Is it possible for a cluster to contain tweets about more than one event? Even if we can design a smart clustering algorithm to guarantee that each cluster represents a single event, how to describe this event based on the data within the cluster?

Two intuitive methods to address the above issues are the summarization [CKP93, HP96, LC05] and the topic modeling [XC99, SSRZG04]. They can be used to summarize the content of the cluster by either sentences or key words/phrases. But both of these two methods fail to address the problem that sentences or key words, which are used to describe the event, may not necessarily appear in every tweet in this cluster. Let us raise an example about delivering interesting events to a customer from the mobile company – Virgin. We identify three key words "Virgin", "mobile", and "signal" from a cluster, there is no guarantee for every tweet in this cluster to contain each one of these three key words. Accordingly, after a key word "signal" is reported to the business customer, he/she is going back to the tweets cluster which is described by "signal" to check what users posted about the signal and realizing that not too many tweets are really complaining about the "signal".

In addition to the clustering method, there are also other methods for the event detection, for example, temporal and spatial models proposed by [SOM10]. These methods require users to define all possible events by themselves at the very beginning. Thus they are less applicable for our event detection problem which expects us to have the ability of detecting new events as new tweets coming.

In order to prevent those issues raised by existing event detection methods and tackle the challenges posed by the event detection from the microblog, we propose a new and simple event detection framework, which is mainly based on the frequency of words in the data, to help the social media audience. To help understand what a event means, we interpret it from two aspects: first, "event" is something **new** according to the historical data; second, "event" could be something appeared before. However, it is shown much more frequently in the tweets currently (we name such "event" anomaly). Based on the interpretation, our framework aims at detecting "new" and "anomalous" events simultaneously as soon as they are worth the attention of the social media audience.

Our contribution in this work can be summarized in three-fold. (1) We propose a new and simple event detection framework to help customers monitor users' feedback reflected in the online microblog service, and report the "new" and "anomalous" events hidden in users' feedback. (2) Our framework has two versions of implementation, including a batch version and a streaming version. With these two versions, one could not only identify events from a given set of documents/tweets at one time, but also detect events from the continuously coming documents/tweets as long as there are new data. (3) Although this framework is tuned by performing several experiments over different compositions of parameters on the real-world tweets for different domains, it is convenient for a social media reader to indicate his/her own parameters based on the domain specific knowledge.

4.1.3 Multi-Task Multi-Label Classification

Sometimes audience are interested in sketching a storyline of a known event. Some other times, they would like to pay attention to new events. What if they are interested in both sentimental and topical information of incoming tweets? How does one help them in this case?

In order to help the above audience to summarize tweets, we focus on two aspect: (1) sentiment of a tweet that captures the subjective mood of a user, such as positive and negative; and (2) topic of a tweet that indicates the scope of subject content from pre-determined aspects, such as Compliment, News, and Promotion. In general, techniques known as sentiment analysis and topic analysis respectively are used to infer latent sentiments and topics of a given text corpus. Furthermore, in this dissertation, we employ the following class schemes. The sentiment classes are positive, negative, and neutral. The topic classes include Care/Support, Lead/Referral, Mention, Promotion, Review, Complaint, Inquiry/ Question, Compliment, News, and Company/Brand. We focus on the problem of classification, i.e., given a set of pre-determined classes, how to identify which classes an instance belongs to.

Given a collection of tweets regarding a certain common subject, a topic classification method can reveal the particular aspects that users are talking about and which aspects are dominant, while a sentiment classification method tells the proportion of users who feel positive or negative toward the subject. The analysis of tweet sentiments and topics can help businesses to get a sense of user opinion towards their products and services. Due to the practical implication, in recent years, a lot of studies (e.g., [12, 21, 1, 24, 19]) have been conducted towards sentiment and topic classifications of tweets (see Section 2 for details).

Time-sensitive OSN contents, such as tweets are usually about many topics, and carrying various sentiments. Since a large number of the tweets are generated continuously, it is hard for an individual to read each of them. Document sum-

marization is definitely a promising method as introduced before. However, it is useful if a single tweet can be assigned a topic label and a sentiment label. In this case, topic and sentiment together are used to summarize a tweet. The problem is how to classify tweets onto topics and sentiments automatically and quickly.

There are a lot of challenges in sentiment and topic classification for tweets. Tweets are usually short, and sometimes composed of incomplete sentences. In the mean time, the social network users tend to write tweets with informal language, which involves many acronyms and slang terms. Furthermore, a single tweet, as the traditional textual document, might contains more than one topic. Therefore, it is sometimes difficult for human being to classify a tweet into a topic. According to our study of tweet-topic assignment by human being for tweet sets, when each tweet is assigned three topic labels by three different people, only less than half of the tweets have three consistent labels. In other words, three people cannot reach an agreement over more than half of the given tweets. As a result, no one can expect the traditional binary classification to produce good results.

Based on the limitations of existing methods, in this work, we propose a model, termed as the Multi-Task Multi-Label (MTML), which performs the classification of both sentiments and topics of tweets concurrently, and incorporates each other's information to reinforce each classification performance.

4.2 Related Work

Several research directions are related to our work, including microblog mining, information retrieval, and multi-document summarization.

4.2.1 Microblog Mining

The emergence of Twitter motivates recent research works on mining microblogs, including microblog search [Efr11], identifying emerging topics on Twitter [MK10], and summarizing tweets in a certain period [TYO11]. A few research

works have been devoted to event detection [SOM10, SKC11], but they focus on the detection of novel events without a global view.

To achieve a better performance, several research methods have been proposed to deal with the unique characteristics of microblogs, e.g. expanding tweets by hashtags [Efr10], utilizing social relations for identifying influential tweets [HBS10], incorporating sentiment categorization [BS10], promoting most recent tweet [EG11], employing transfer latent topic models for overcoming abbreviated texts [ZLLC11], and expanding queries by recently frequently co-occurred terms [MTdRW11].

The dynamic and social nature of microblogs is not fully explored by previous research efforts. By adding a temporal dimension in the event storyline generation system, our work sheds light on the understanding and mining of microblogosphere.

4.2.2 Text Summarization and TDT

Multi-document summarization conveys the main and most important meaning of several documents. One type of summarization systems select representative sentences, e.g. with significant frequency [YGVS07], or structural centroid in sentence graph [LLL11, KMS04]. Another type is based on matrix decomposition [LS00, WLZD08, SM00]. Some prior researches focus on clustering query-induced results [WLLH08].

Not until recently, a limited number of studies devote to summarizing documents with time stamps, mostly news articles. For example, in [MZ05], an HMM style model is presented to discover evolutionary theme patterns (term distributions). BlogScope [BK07] discovers hot trend and temporal keyword correlations. Similarly, a burstness-aware search framework is presented in [LAP+09]. A finite mixture model is presented in [MY04] for tracking dynamics of topic trends. In [LL08] a main theme is extracted by selecting representative sentences in each time segment. ETS [WZLD09] returns the evolution skeleton along the

timeline by extracting representative and discriminative sentences at each phase. In [YWO+11] representative sentences are chosen based on relevance, coverage, coherence and cross-date diversity. In [TYO11] summarization consists of median tweets in each time segment.

Unlike multi-document summarization, research in the area of topic detection and tracking (TDT) [All02] aims to thread streams of texts. Most works along this direction are devoted to clustering and classifying similar texts, without considering the timestamps of articles, e.g. relevance model in [LAD+02] adopts symmetric similarity comparison. Others consider the influence among articles to be unidirectional and directly dependent, e.g. topic structure is identified in [MY04] by forgetting out-of-date statistics, the bursty structure is recognized in [Kle03] by estimating state transition probability in an infinite state automation.

To conclude, although these methods have been successfully applied in their own domains, they are not applicable to this storyline generation problem. The quality of generated storyline is determined by the quality of summary in each phase, and the quality of phase segmentation. However, the asynchronism of information propagation in the microblogosphere makes it difficult to partition the timeline of an event into different phases. Therefore, previous summarization and TDT methods lack the ability to generate a complete and coherent storyline. On the contrary, our graph optimization based method has a built-in mechanism to simultaneously generate the summary for each virtual phase and naturally integrate the generated summaries to form the storyline.

4.2.3 Event Detection

Event detection is a popular problem which involves many real-world applications, such as surveillance, scientific discovery, and fault detection.

Event detection is useful and important because on the one hand, it extracts the most latest events of interests, summarizes, and presents them to the users; on the

other hand, it involves many real world applications, such as surveillance, scientific discovery, and fault detection. For example, it could detect the disease outbreaks before the situation turns to quite severe [WTE+01], so that large amounts of cost in both lives and dollars can be reduced. For another example, by injecting the event detection into the environment monitoring [Der07], the spike or the anomaly of the temperature can be observed and cleaned as soon as it appears, thus researchers could have cleaned data for their research purpose.

4.2.4 Topic and Sentiment Classification

Tweet sentiment and topic analysis becomes very popular recently. However most state of the art studies address only sentiment classification or topic classification. To determine tweet sentiment, query-based dependent features and related tweets are explored and incorporated in [JYZ+11]. In [AXV+11], POS-specific prior polarity features are introduced and applied with a tree kernel for sentiment analysis. Tan et al. find that including the influence of social connections can improve accuracy of sentiment classification [TLT+11]. In addition, a graph model is introduced to classify sentiment of hashtags in a time period [WWL+11].

To classify topics of noun phrases in tweets, a community-based method is presented to identify their boundaries within the context and classify them to a specific category [CCBL12]. After that, a model that switches between two probability esti- mates of words is proposed, which can learn from stationary words and also respond to bursty words [NHF12]. In [NBFH11], another method is introduced to determine whether a tweet is related to a topic or not by using data compression. Furthermore, a Bag-of-Words approach and a network-based approach are evaluated in classifying twitter trending topics into 18 general categories [LPN+11]. These approaches focus on single-label classification on either sentiment or topic classes. Among the state-of-the-art work, none of them studies multi-label classification that analyzes both sentiments and topics simultaneously.

4.3 Storyline Generation

We first briefly introduce the architecture of the storyline generation. As discussed in the related work of this chapter, to generate the storylines from relevant tweets, obstacles are duplicated tweets and indirect retweets. Intuitively, we can pick up a good tweet to represent similar or duplicated tweets. The representative tweets provide the basic outline for each phase. Then the representative tweets are connected appropriately to depict the evolving structure of the event. In order to eliminate noisy retweets, only texts published after a certain time can be considered as subsequent phases. Finally, there may be different ways of connecting these representative tweets, and an optimistic connection should be the one that connects them most smoothly.

The storyline generation procedure consists of three parts. In the first part, a multi-view tweet graph is constructed, in which the semantic and temporal information among relevant tweets is stored. Next, representative tweets are extracted by finding a minimum dominant set on the tweet graph. Finally, a minimum steiner tree algorithm is employed to connect the representative tweets in each phase.

Given an event query Q and a collection of relevant tweets by the method described in the work [LLL+12b], we can construct a multi-view tweet graph.

Definition 4.3.1 (multi-view tweet graph) *A multi-view graph $G = (V, W, E, A)$, where V is a set of vertices (nodes), W is the weights of V, E is a set of undirected edges, which represents the similarities between tweets, and A is a set of directed edges (arcs), which represents the time continuity of the tweets.*

Construction of such a graph is controlled by three nonnegative real parameters α, τ_1, τ_2, $\tau_1 < \tau_2$. Each node in G represents a tweet. We use the cosine measure to calculate similarity between two tweets. To define E, we join the two nodes by an edge if and only if the text similarity between the two responding tweets is greater than α. To define A, we draw an arc from v_i to v_j if and only if $\tau_1 \leq t_j - t_i \leq \tau_2$,

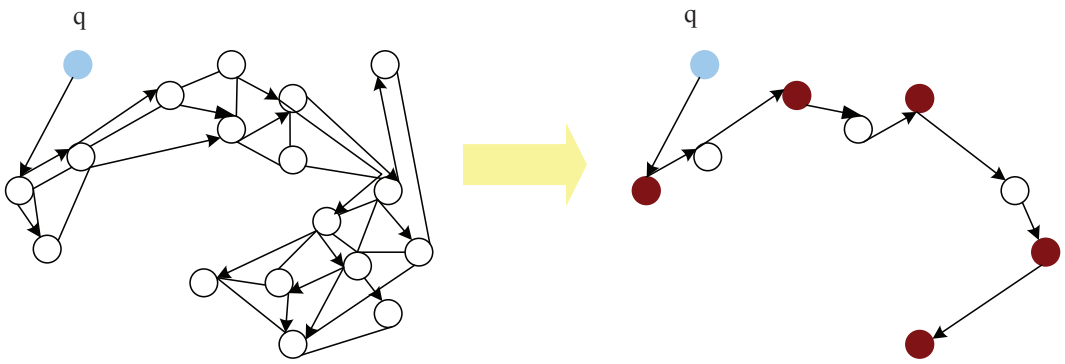


Figure 4.2: An illustration of the storyline generation.

where t_i and t_j are their respective time stamps. We call $[\tau_1, \tau_2]$ the temporal window. Also, for each node v_i, its vertex weight, $w(v_i)$, is $1 - score(Q, v_i)$. In our method, we first find the dominating set on the undirected graph $G = (V, W, E)$ (i.e., without considering A in the multi-view graph), and then perform the steiner tree algorithm to connect the dominating set on the directed graph $G = (V, W, A)$ (i.e., without considering E in the multi-view graph) which takes the time continuity into consideration and leads to a coherent storyline.

A subset S of the vertex set of an undirected graph is a dominating set if for each vertex u, either u is in S or is adjacent to a vertex in S. The problem of finding a set of representative summaries can be viewed as the minimum-weight dominating set problem on the undirected graph (V, W, E).

Definition 4.3.2 (MWDS) *The Minimum-Weight Dominating Set Problem (MWDS) is the problem of finding, given a vertex- weighted undirected graph G, from all dominating sets of $G = (V, W, E)$, the one whose total vertex weight is the smallest.*

We consider the following straightforward greedy algorithm for obtaining an approximate solution (Algorithm 6). This algorithm views that the weight of a newly added vertex is evenly shared among its newly covered neighbors and selects the node that minimizes this share at each round of iteration. The approximation rate

of this algorithm is $1 + \log(\Delta \|OPT\|)$, where Δ is the maximal degree of G and OPT is the optimal dominating set.

Algorithm 6 Greedy $MWDS$ Approximation

Input: $G = (V, W, E), m$ (*maximum number of items in the dominant set*)
Output: dominant set S

$S \leftarrow \emptyset, T \leftarrow \emptyset$;
while $|S| < m \&\& S \neq V$ **do**
  for $v \in V - S$ **do**
    $s(v) = \|\{v' | (v', v) \in E\} \setminus T\|$;
    $v* = \arg\min_v \frac{w(v)}{s(v)}$;
    $S \leftarrow S \bigcup \{v*\}$;
    $T \leftarrow T \bigcup \{v'' | (v'', v*) \in E\}$;
  end for
end while

Once we select the most representative summary in each phase using the dominating set approximation, we need to generate a natural storyline capturing the temporal and structural information of the event-relevant tweets. To study this problem we use the concept of Steiner trees. Here a Steiner tree of a graph G with respect to a vertex subset S is the edge-induced sub-tree of G that contains all the vertices of S having the minimum total cost, where the cost is the total weight of the vertices.

Definition 4.3.3 (Steiner Tree) *Given a directed graph $G = (V, W, A)$, a set S of vertices (terminals), and a root $q \in S$ from which every vertex of S is reachable in G, find the subtree G rooted at q containing S with the smallest total vertex weight.*

The problem is known to be NP-hard since the undirect- ed version is already NP-hard. A straightforward solution for this problem is to find the shortest path from the root to each of the terminal and merge the paths. Of course, combining lightest paths does not guarantee the minimum total cost. Consider an extreme case in which there is a cost C_{opt} from the root to a vertex v in the graph and a zero cost path from v to each terminal, and there are paths of cost $C_{opt} - \epsilon$ from

the root to each terminal. The total tree cost when v is used as an intermediate vertex is C_{opt}, but the total cost is $k(C_{opt} - \epsilon)$ when the straightforward solution is used.

Algorithm 7 Steiner Tree Algorithm

Input: $G = (V, W, A), S, q, k \geq 1$
Output: Steiner tree T rooted at q covering at least k vertices in S

$T \leftarrow \emptyset$;
while $k > 0$ **do**
  $T_{best} \leftarrow \emptyset$;
  $cost(T_{best}) \leftarrow \infty$;
  for $v \in V, (v_0, v) \in A, 1 \leq k' \leq k$ **do**
    $Tp \leftarrow A_{i-1}(k', v, S) \bigcup \{(v_0, v)\}$;
    if $cost(T_{best}) > cost(T')$ **then**
      $T_{best} \leftarrow T'$;
    end if
    $T \leftarrow T \bigcup T_{best}$;
    $k \leftarrow k - \|S \bigcap V(T_{best})\|$;
    $S \leftarrow S \setminus V(T_{best})$;
  end for
end while

This observation suggests Algorithm 7. The initial call of $A_i(k, q, S)$ with S set to the dominating set calculated by algorithm 7, q set to be event vertex assigned with the earliest time stamp, and k set to be the size of S. The algorithm takes a level parameter $i \geq 1$. $i = 1$ is the default case where the straightforward algorithm selects l vertices closest to root and returns the union of the shortest paths. The length of an arc $(u, v) \in A$ is the vertex weight of u. We will interpret the output tree as the storyline transition from the root to all the other dominating objects as illustrated in Figure 4.2. For a constant i, the algorithm is known to run in polynomial time and produce an $O(k^{1/i})$ approximation.

4.4 Event Detection

4.4.1 Framework

An event can be described by a sentence, a phrase, or even a word. In this dissertation, the data set input to the framework is collected from the Twitter. Thus, the detected event from the data set is always the sentence, phrase, or word in the tweet.

As introduced before, our framework has two versions of implementation, (1) the batch version, which deals with a set of tweets at one time; (2) the streaming version, which continuously processes the incoming tweets. Before presenting these two version, it is worth introducing several concepts.

First, as our framework is trying to detect events from tweets and report them to audience, so that audiences can pay attention to their interested events. According to the suggestions by domain experts, we noticed that not every event is important to customers, the event does matter is really the one appears in the tweet with negative sentiment. A tweet with negative sentiment means a tweet presenting its author's bad emotional states, such as angry, sad, etc. On the other hand, the customer may not care that much for the tweet with positive sentiment. A tweet with positive sentiment denotes a tweet which shows its author's good emotional states, e.g., happy. Now that we are clear about the tweet with positive/negative sentiment, the "positive"/"negative" term/document frequency of a noun or a phrase composed of nouns can be defined accordingly. The positive/negative term frequency of a noun or a phrase composed of nouns indicates the number of times this noun/phrase appears in the tweet with positive/negative sentiment. The positive/negative document frequency of a noun or a phrase indicates the number of tweets contain this noun/phrase.

Second, besides the sense that events with negative sentiment should be reported to the audience, not all events have to be delivered to them if the number of detected events is really huge. Then how to decide which event to be reported?

We want to make sure the reported events are really important. A traditional approach to define the importance of a term in the document set is TFIDF, which is usually given by:

$$TFIDF(term, doc, Docs) = \sum_{doc \in Docs} tf(term, doc) \times idf(term, Docs), \quad (4.1)$$

where tf denotes term frequency, while idf indicates inverse document frequency. In this definition, the importance of a term is composed of its frequency in every document. Since each document has its own time stamp, e.g., the time stamp when this document was posted on the web sites, we suggest that the importance of the term frequency in a particular document will decay as time goes on. Hence, the time decay information is adopted into the definition of TFIDF as:

$$TFIDF(term, doc, Docs) = \sum_{doc \in Docs} (tf(term, doc) \cdot decay) \times idf(term, Docs), \quad (4.2)$$

In the following sections, whenever we mention TFIDF, it is TFIDF in Equation 4.2.

Third, "New" event detection is the task, which intends to identify those events, which have not been seen before and have large number of appearance in tweets currently. To be specific, if there is a topic with a specific semantic meaning and carries the negative sentiment which is never mentioned by users in Twitter system and suddenly attract much tweeting traffic, this topic is worth being tagged as "New" and read. Moreover, we can explain it with an example. Suppose we found an event "signal", which never shows in the historical data collected for the Mobile service company, in the latest tweets with negative sentiment, and this event appears in more than n (n is the predefined threshold) tweets, it will be marked as "New".

Forth, "Anomalous" event detection is the task, which aims at identifying those events, which suddenly have a very large number of appearance even though it has ever shown before. In order to perform anomalous event detection, both

two versions of our framework rely on the Grubbs' test [oSUC+01] which is originally proposed for detecting the outlier. For a specific event represented by a noun/phrase, supposing that the input to the test is the negative term frequency within different time windows, Grubbs' test can assist us to find out the "outlier" negative term frequency within one or more time windows. Then we claim that this event is anomalous in those time windows.

4.4.2 Batch Version

Sometimes, audience are interested in figuring out what events are in the data set they collected before. For this case, our framework has to process all tweets within the data set at one time and produce the results for the customer, thus, we call this processing procedure – "Batch processing".

The batch processing procedure are described in Algorithm 8:

Algorithm 8 Batch version

Given A – the set of tweets with the corresponding sentiment in the data set, t – the time window for detecting anomalous event.

1. Perform the sentence splitting, tokenizing, stemming, POS tagging for each sentence in tweets in A.

2. Clean the preprocessed tweets by removing stop words, and dirty words.

3. Extract all nouns and noun phrases.

4. Compute TFIDF score for each noun or noun phrase, rank them by their TFIDF.

5. Obtain the negative term frequency within every possible t (according to the time range indicated by the first and the last tweets of the data set) for each noun or noun phrase.

6. Detect the anomalous noun/phrase using Grubbs' test based on the data from step 5.

4.4.3 Streaming Version

In the previous subsection, we present the batch version of our framework. However, in most cases, the audience would like to continuously detect events instead of detecting only once. The general procedure of the streaming detection is sketched in Algorithm 9.

Algorithm 9 Streaming version

Given a set of historical tweets with the corresponding sentiment, t – the time window for detecting events, T – the historical window, ut – the time window for updating the historical nouns/phrases, hn – the number of top ranked nouns/phrases from the historical data, tn – the number of top ranked events for detecting anomalies.

Read tweets within the historical window T into memory.
if FirstTimeStart **then**
  Initialize the top hn nouns/phrases
else
  Update the top hn nouns/phrases
end if
while NewTweetsComing **do**
  Read tweets within the latest ut into memory.
  Update the top hn based on the new tweets in the memory.
  Detect new events
  Detect anomalous events
end while

To fully understand this streaming algorithm, an example is given in the Figure 4.3.

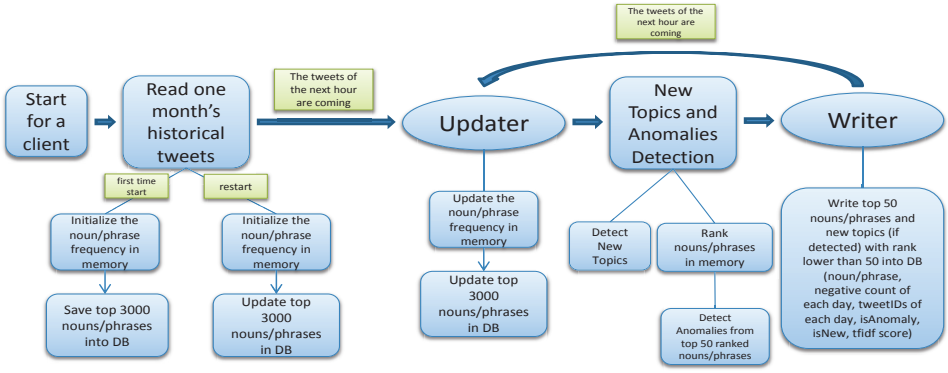


Figure 4.3: An example of the streaming algorithm.

4.5 Multi-Task Multi-Label Classification

4.5.1 Problem Statement

Topics and sentiments are never completely independent for tweets. By reading the tweets, which receive the consistent topic label and sentiment label, we do find some associations between topics and sentiments in terms of the frequency for one topic and one sentiment are assigned to tweets. Meanwhile, some terms/phrases in tweets are strong indication of the association between some topics and sentiments, e.g., the term — "Love" usually denotes the association between a topic — "Compliment" and a sentiment — "Positive", and the term — "Crap" is considered as the bridge between a topic — "Complaint" and a sentiment — "Negative".

When an audience hopes to quickly browse through a large set of tweets, the presentation of the topics and sentiments for this set would be helpful. Therefore, we consider using the Multi-Task Multi-Label (MTML) classification to address the "summarization" problem of tweets for audience. With this MTML solution, several benefits can be listed below:

1. Performing two tasks, including topic classification and sentiment classification, simultaneously.
2. Utilizing the sentiment information to strengthen the performance of topic classification via the topic-sentiment association.
3. Providing an alternative summarization instead of the sentence-extraction-based summarization and event detection.

Formally, the multi-task multi-label (MTML) classification is defined as follows:

Problem 1 (MTML Classification) Given an instance x and classification tasks $T = \{T_j : j = 1, ...M\}$, where the j-th classification task T_j has a finite set of classes $L_j = \{l_{jk} : k = 1, ...Kj\}$, the goal of MTML classification is to find a

collection of class label sets $Y = \{Y_1, ...Y_j...\}$ that x belongs to, $Y_j = \{l_{j1}, ...l_{jq}\} \subseteq L_j$ is the set of class labels of x for the j-th classification task.

Two questions need to be investigated before designing the solution to the MTML problem.

1. How does one make use of multi-task classification to reinforce each task?
2. How does one incorporate and process multiple labels in multi-task classification?

To answer the above two questions, two strategies are proposed to address the MTML problem.

1. Given the topic labels and sentiment labels, as well as the motivation to associate every pair of topic and sentiment labels, a new label set is produced by combing every topic and sentiment labels. For example, an instance x has one topic label l_{ip} and one sentiment label l_{jq}. A new label $l_{ip} - l_{jq}$ combing the topic label and sentiment label can be generated for x as a single label.
2. Two classification tasks are performed separately. In order to incorporate the information of topic/sentiment into the task of sentiment/topic classification, the original features of the instance for topic/sentiment classification are extended by one more feature, which carries the information of sentiment/topic.

Since the latter strategy cannot help perform two classification tasks at the same time, the former one is preferred. It is worth noticing that there is an assumption that each data instance is associated with only one topic label and one sentiment label without any ambiguity. However, as discussed before, it is nontrivial to assign a classification label to a tweet, and different people might have different opinions about the label assignment for one tweet. Due to this problem, we design the MTML solution as follows:

1. Define a topic pool (contains multiple topics) and a sentiment pool (usually contains three label, including "Positive","Negative", and "Neutral").
2. Invite n workers to read tweets, and then manually assign a topic label (from the topic pool) and a sentiment label (from the sentiment pool) to each tweet according to their own understanding. Thus, each tweet contains n topic labels and n sentiment labels.
3. Apply the first strategy for the MTML problem to generate new label for each tweet. Specifically, given a tweet, combine each one of n topic labels and each one of n sentiment labels to form a new label, and then this tweet will have n^2 new labels. After, this tweet is duplicated $n^2 - 1$ times. And each one of the n^2 copies is associated with one of the n^2 new labels. Finally, these n^2 new data instances of tweets are put back to the data set.
4. Utilize one of the well-accepted classification methods (e.g., SVM, Maximum Entropy, etc.) to train a classifier.
5. Given a new tweet, classify it using the classifier obtained from the last step. And then extract the topic and sentiment from the classification label.

4.6 Experiments for Storyline Generation

In the experiments, we evaluate the performance of the proposed storyline generation method. In particular, we compared this method against several well-known multi-document summarization, and our summarization framework in Chapter 3. We also conduct a user study to compare our method with different document understanding systems.

4.6.1 The Data Set

The data set is Tweets2011 corpus for TREC 2011 microblog track. The corpus is comprised of 2 weeks (23th January 2011 until 8th February) of sampled tweets from Twitter. Different types of tweets are presented, including replies and

retweets. The corpus is multilingual, including English, Japanese and so on. More details of the collection are illustrated in Table 4.1.

Number of tweets	15137399
Number of English tweets	9318772
Number of retweets	1487299
Number of English retweets	1069006
Number of users	4670516
Median Tweet Length	8.66
Median English Tweet Length	10.76

Table 4.1: Statistics of Data set.

In pre-processing, we do not remove stop-words. Instead, mentions (@somebody) are removed from the vocabulary. Non-English tweets containing less than one English word with more than 2 characters are filtered. Explicit re-tweets with HTTP code 302 are filtered. Empty tweets and forbidden tweets with HTTP code 403 and 404 are also filtered. Porter stemmer is adopted in indexing.

4.6.2 Summarization Capability

Note that after retrieving the relevant tweets, various document summarization methods can be adapted to form the storyline by extracting the most relevant tweets. In this section, we conduct experiments to compare the summarization performance of different approaches including our proposed one, aiming to show the advantages of using the Dominant Set and the Steiner Tree to generate the storyline from the summarization aspect.

The measurement used in this subsection is mainly based on Recall-Oriented Understudy for Gisting Evaluation (ROUGE) – an evaluation toolkit for document summarization [Lin04] which automatically determines the quality of a summary by comparing it with the human generated summaries through counting the number of their overlapping textual units (e.g., n-gram, word sequences, and etc.). In particular, F-measure scores of ROUGE-2 and ROUGE-SU4 are presented for our experiments. 49 queries provided by TREC 2011 microblog track are used in

the experiments. For each query, first, DPRF is utilized to retrieve the top 1,000 tweets, then 8 students are invited to manually generate the "storyline" (50 tweets are selected) from these 1,000 tweets as the ground truth.

Comparison on Different Summarization Approaches

We compare our method with several well-known and recent summarization approaches including:

1. Random: randomly selects the sentence as the summary;
2. MostRelevant: picks up the sentences which are most relevant with the topic as the summary;
3. Latent Semantic Analysis (LSA): identifies semantically important sentences by conducting latent semantic analysis;
4. K-means: performs K-means over the sentences, then treats centers of all sentence clusters as the summary;
5. Non-negative Matrix Factorization (NMF) [LS00]: performs NMF on the sentence-term matrix and selects the high ranked sentences.
6. Symmetric Non-negative Matrix Factorization (SNMF) [WLZD08]: calculates sentence-sentence similarities by sentence level semantic analysis, clusters the sentences via symmetric non-negative matrix factorization, and extracts the sentences based on the clustering result;
7. Spectral Clustering with Normalized Cuts (NCut) [SM00]: performs the Spectral Clustering using Normalized Cut to cluster the sentences, and then uses centers of clusters as the summary;
8. Query-sensitive Mutual Reinforcement Chain (Qs-MRC) [WLLH08]: extends the mutual reinforcement principle between sentence and term to document-sentence-term mutual reinforcement chain, and uses query-sensitive similarity to measure the affinity between the pair of texts;

Methods	ROUGE2	ROUGE-SU
Random	0.0425	0.0903
MostRelevant	0.0526	0.1075
LSA	0.0403	0.0857
K-means	0.0489	0.1002
NMF	0.0534	0.1043
SNMF	0.0593	0.1203
NCut	0.0635	0.1156
Qs-MRC	0.0647	0.1255
MSSF	0.0639	0.1324
DS Only	0.0731	0.1280
DS+ST	0.0895(++)	0.1363(+)

Table 4.2: The comparison among different summarization methods. Notice that DS denotes Dominant Set, and ST represents Steiner Tree. ++ and + indicate that DS+ST significantly outperforms the best comparative methods with a confidence level greater than 99% and 95%, respectively.

9. Multi-Document Summarization using Submodularity (MSSF) [LLL11]: a multi-document summarization framework based on Submodularity;

10. Dominant Set (DS only): Document summarization using the Dominant Set algorithm (i.e., Algorithm 1 in Section 4.3).

The comparison of our proposed method (DS+ST) with other summarization methods is presented in Table 4.2. It can be seen from the results that our proposed DS+ST outperforms all the other summarization methods. In addition to the comparison of DS+ST against the other summarization methods, we employ the standard t-test to determine whether the performance improvement of DS+ST over the others is statistically significant. The results show that the improvements of our DS+ST on both ROUGE2 and ROUGE-SU are significant.

The good results of our method benefit from the following two aspects. (1) The Dominant Set algorithm (i.e., Algorithm 1) used in our method can select tweets which are similar to both the given query and all the other tweets. Thus it is not only good at extracting the representative information from the given sentences to form a reasonable summary, but also providing an appropriate mechanism to select the "dominant" nodes to generate storylines. (2) The Steiner Tree algorithm (i.e.,

Algorithm 2) is capable of detecting the "outline" of all the given sentences from the dominant nodes. Thus comparing with the other traditional summarization methods, it is able to generate more natural and logical storylines/summaries.

As a result, by combining the Dominant Set algorithm and the Steiner Tree algorithm, our proposed method is suitable for generating the "storyline" from the messages delivered by microblog services.

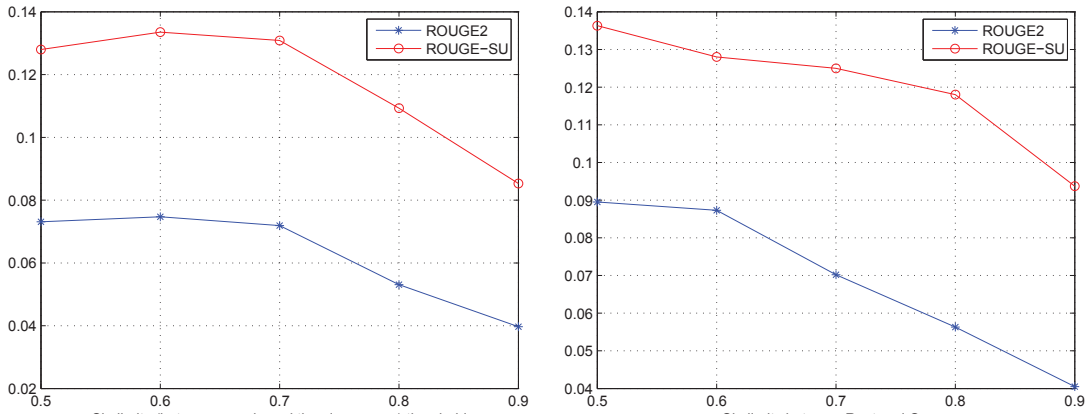
Parameter Tuning

In addition to the above comparison with different methods, we further study the summarization results by tuning the parameters of the Dominant Set algorithm and the Steiner Tree algorithm.

First of all, we study the Dominant Set algorithm by varying the "similarity threshold". We vary the threshold for the similarity between each tweet and the given query from 0.5 to 0.9 with a step size of 0.1 (totally 5 steps).

Secondly, one may notice that a key step before performing the Steiner Tree algorithm is to pick up a ***root*** node. A good root node could start a good story from tweets. In general, a good root should satisfy two conditions: 1) it should start as early as possible in terms of the post date of the tweet; 2) it should be similar to the given query. Usually, we choose the earliest node within the Dominant set as the root. However, we also study how the "similarity to the given query" influences the final summarization results. In other words, the earliest node may not necessarily be the root, but a later node from which every node of the Dominant set is reachable in graph G can be the root as long as it is more similar to the given query. To choose the root, we vary the similarity to the given query from 0.5 to 0.9 by a step size of 0.1. The comparison results by tuning the parameters are shown in in Figure 4.4a and 4.4b.

We have two observations from Figure 4.4a. (1) The selection of the similarity threshold does influence the summarization performance of the Dominant Set



(a) (b)

Figure 4.4: (a) Similarity (between a node and the given query) threshold; (b) Similarity between Root and Query

algorithm. An inappropriate similarity threshold may weaken the Dominant Set greatly. (2) It is hard to claim that a larger similarity threshold would result in a better performance. In fact, when the similarity threshold is greater than 0.6, the summarization performance decreases as the threshold increases. The intuitive explanation is that a large similarity threshold may induce the algorithm to omit some important tweets which are not similar enough to the given query.

The observation from Figure 4.4b is that as the similarity to the given query increases, the summarization performance on both ROUGE2 and ROUGE-SU keeps going down. By analyzing it, we find that a large similarity threshold could lead to a "late" root. For example, a "late" root may exactly match the query, however, it would start the story from the middle of the whole storyline. In such a case, the tweets before the storyline's middle point are omitted, thus the evolving structure of the storyline is not well maintained.

4.6.3 A User Study

Since storyline generation is a subjective process, to better evaluate the retrieved tweets and the generated storylines, we conduct a user survey. The subjects of the survey are 18 students at different levels and from various majors of a

research university. In this survey, we randomly sample 10 queries and 500 English tweets. Each participant is asked to read these tweets and 3 queries, and compare the results of different systems in a random order from the following point of views: relevance, coverage, coherence, and overall satisfaction. A score of 1 to 5 needs to be assigned to each system according to the user's satisfaction of the results. A rank of 5 (or 1) indicates that the result of the system is the most (or least) satisfactory. We implement the following systems for comparison.

1. Top10-Recency: presents the top 10 retrieved tweets by the recency language model RLM on the original queries.
2. Top10-DPRF: presents the top 10 retrieved tweets using the DPRF query expansion [LLL+12b].
3. RecencySum: performs document summarization based on the retrieved tweets using the recency language model. MSSF is used as the document summarization method since it obtains the best results in Section 4.6.2.
4. DPRFSum: performs MSSF based on the retrieved tweets using the DPRF query expansion.
5. RecencyTimeline: generates timeslines [YWO+11] based on the retrieved tweets using the recency language model.
6. DPRFTimeline: generates timelines based on the retrieved tweets using DPRF query expansion.
7. RecencyStoryline: generates storylines using the methods proposed in Section 4.3 based on the tweets retrieved by the recency language model.
8. DPRFStoryline: generates storylines based on the tweets retrieved by DPRF query expansion.

Table 4.3 shows the user rated scores for each system. From the results, we have observations as follows. (1) The performance of tweet retrieval is critical. The proposed DPRF query expansion approach outperforms the recency language

	Relevance	Coverage	Coherence	Overall
Top10-Recency	3.06	1.67	1.50	2.06
Top10-DPRF	3.39	1.83	1.56	2.28
RecencySum	2.94	2.33	2.39	2.72
RecencyTimeline	3.06	3.06	2.83	3.33
RecencyStoryline	3.06	3.78	4.00	3.78
DPRFSum	3.22	2.50	2.44	3.05
DPRFTimeline	3.33	3.33	3.06	3.83
DPRFStoryline	**3.39**	**4.17**	**4.28**	**4.12**

Table 4.3: Survey Results: User ratings on different systems based on their satisfaction.

method. (2) Although the listed top 10 query results are highly relevant to the query, there also exists high redundancy among the top-ranking query results, thus the coverage and coherence of the results are poor. (3) Summarization based results achieve higher overall satisfaction than the methods of listing top query results because it can help users better understand the tweets. (4) Users prefer structured results such as timelines and storylines than pure text summaries. (5) The proposed storyline generation methods outperform the timeline generation method because the structures contained in the storylines can assist users quickly grasp the event evolution.

4.7 Experiments for Event Detection

In this section, our goal is to evaluate our event detection framework on the tweets crawled for specific commercial brands of business customers, i.e., Sprint-Mobile, Crest, and Holiday-Inn. Note that our event detection framework is way different from the previous work as we introduced before, therefore, the traditional evaluation metrics and comparison methods of the event detection cannot be applied for our experimental purposes. Instead, we conduct the preliminary experiments over the commercial brand related tweets to show the detection capability of the framework.

Brands	# negative tweets	# of non-negative tweets
Sprint-Mobile	1,928	6,462
Crest	1,593	15,209
Holiday-Inn	2,339	41,577

Table 4.4: The description of the data set.

4.7.1 The Data Set

The event detection in this dissertation heavily depends on the microblog documents, because the microblog is known as a new kind of media which shows better performance in terms of the ability of capturing the on-going events. The problem with microblog documents is that they cover a extremely wide range of topics and prevent users in a specific domain from easily extracting domain-specific events from those about different topics. Therefore, in order to evaluate the capability of reporting the domain-specific events of our event detection framework, we crawl microblog documents (specifically, tweets from Twitter) for three specific brands – Sprint-Mobile, Crest, and Holiday-Inn via the API provided by Twitter. The statistics information of the tweets are in Table 4.4.

4.7.2 Technical Set Up

Our framework is implemented in Java and deployed on the server with Intel Core i5 CPU (2.40 GHz) and 8 GB memory. In the meantime, another server is employed to continuously crawl the tweets for three above mentioned brands and save them into the Mysql database hosted locally. Thus, this crawling server is responsible for collecting tweets, while the framework on the deployment server is in charge of performing event detection over the tweets on the crawling server. It is worth noticing that the well known toolkit – GATE [Cun02] is utilized for tweet preprocessing.

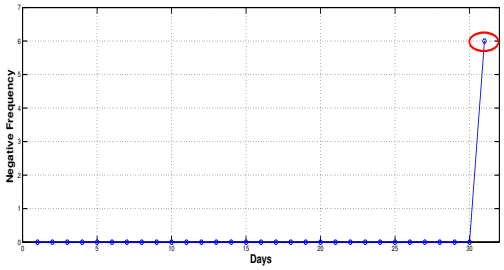
The batch version of the framework is simple to set up. Users could indicate a specific time range and a specific brand. Upon the indication, the batch version performs event detection over the brand specific tweets within that time range at

one time. Considering the suggestions given by domain experts, the data of the first 30 days is used for training – summarizing the normal negative frequency of a noun or a noun phrase; the data of each of next days is used for testing – detecting the abnormal negative frequency (too high comparing with the training data in the first 30 days); it is worth noticing that even the frequency of 1 would be high if all historical frequencies are 0, thus a frequency threshold (in this case, it is set as 5) is required in determining if a particular negative frequency is high; only the top 50 nouns/phrases can be treated as the candidates of events which should draw customers' attention.

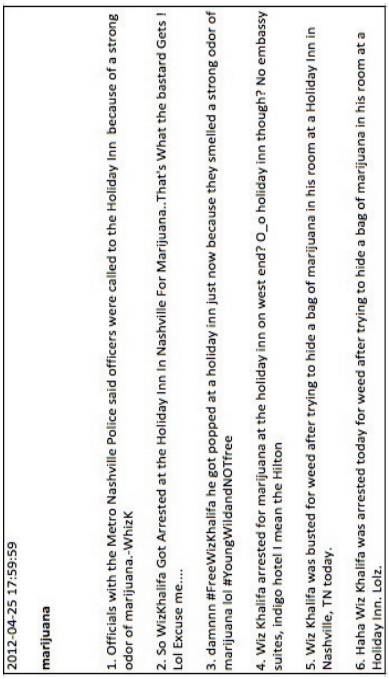
The set up of the streaming version is more complex. Advised by the domain experts, the tweets of the first 30 days are used for training; only the top 3,000 nouns/phrases in the historical window (30 days) are maintained for detecting events. For every other hour, the top 3,000 nouns/phrases would be updated based on the new coming tweets within that hour. In the event detection procedure, the tasks of "new" and "anomalous" event detection are executed every 24 hours: for the "new" event detection, if a specific event does not show in the last 60 days, and it appears more than 3 times within the latest 24 hours, we mark it as "new"; for the "anomalous" event detection, if a specific event appears more than 5 times within the latest 24 hours and is considered to be abnormal via Grubbs' test, it is marked as "anomalous". Note that the numbers 3 and 5 here are thresholds for "new" and "anomalous" event detections respectively. Since "anomalous" event detection has a more strict threshold than "new" event detection, an event could be "new" firstly, then "anomalous" later on.

4.7.3 Detection Results

In this section, two example events which represent the detection schema of the framework are first presented in Figure 4.5 and 4.6, representing new and anomalous event respectively. Then the "new" and "anomalous" events for three

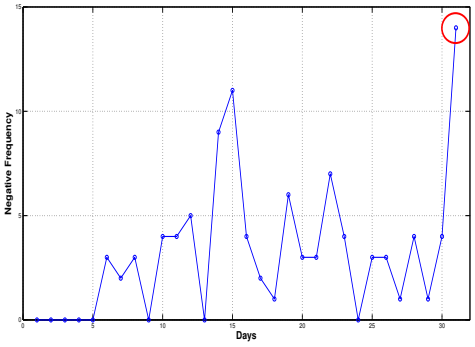


(a) Negative term frequency of each day

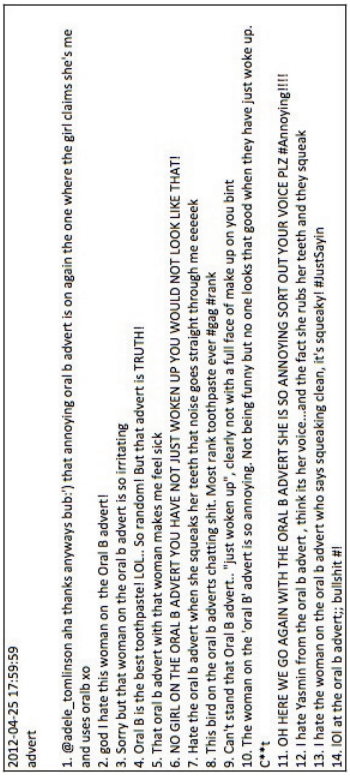


(b) The associated tweets of the 31st day

Figure 4.5: A sample new event – Marijuana.



(a) Negative term frequency of each day



(b) The associated tweets of the 31st day

Figure 4.6: A sample anomalous event – Advert.

brands are presented in Figure 4.7. Note that the core of detection concept of both batch and streaming versions are similar, so the detection results focus on the streaming version, even though the detection results from the batch version for Sprint-Mobile are shown in the Figure 4.8 as an example.

In order to show the detection results of our event detection framework, two example events are first presented in Figure 4.5 and Figure 4.6 respectively. Both Figure 4.5 and 4.6 are composed of two parts: the left part presents the changes of negative term frequency of the event within the 31 days; while the right part shows the associated tweets content for the 31st day, reporting what people mentions about this event during the 31st day. The combination of these two parts gives the root cause of raising this event as "new" or "anomalous".

One can observe from the Figure 4.5 that the event – marijuana is suddenly discussed by many people in Twitter about Holiday Inn because of the arrest of a celebrity in a Holiday Inn hotel during the 31st day of the data set, and since it is never mentioned before, it should be marked as "new". Meanwhile, by observing the Figure 4.6, it can be concluded that the event – advert is keeping being discussed by users in Twitter system, however, because of the number of tweets complain about the advert reaches the limit of naming it as "normal", it is marked as "anomalous".

In order to have a big picture about the detection results, those detected events via the streaming event detection for each of the three brands are reported in Figure 4.7. In this Figure, one can see that the detected events from 2012-04-21 to 2012-04-26 are listed, and those events illustrate what are trending in Twitter about a particular brand. Through reading about these events and their associated tweets, users can quickly capture where the complaints come from.

In addition to the streaming event detection results, a set of detected events by the batched event detection for Sprint-Mobile are presented in Figure 4.8 to show the capability of the batched processing. The results from the batched event

	Sprint-Mobile	Crest	Holiday-Inn
New Events	2012-04-21 virgin_mobile_peep belmont virgin_mobile_network	2012-04-21 match food mouth_kyle	2012-04-21 sale toy realtalk holiday_inn_johnstown themosthighgod bed_bug_bite evictionpartyinmyroom bean holiday_inn_paris_bastille
	2012-04-22 cripple upgrade phone_charge log data_access android_phone	2012-04-22 package yolo crest_fight_plaque tesco_finest crack kit wax crest_mouth_wash crest_sry	sun yolo bevvy bill gurnee wycombe holiday_inn_madness hock landscape voyage peacock crawfish_festival louse holiday_inn_club_vacation_orange_resort
	2012-04-23 boon roaming	2012-04-23 reservation competition commercial mystery directioner mango hoop hammersmith_crest tone bass earth sudden scum drink shop beauty	2012-04-22 pun trk_chilling chelmsford brand_holiday_inn marathon anw logo security train_station maple travel_lodge bath hotel_bathtub holiday_inn_tub
	2012-04-24 screen instruction facebook_account felon technology trip phone_ring tyler	2012-04-24 enamel monger rembrandt_teeth_whitening_kit blend apple tea cracked_crest powerbrush buck rinse barett clip shot uble	2012-04-23 brentwood marketing_mail catering holiday_inn_broadband joburg bedford staff_response broadband obligation sheet recipe steal
	2012-04-25 tyler peak hood insanity virgin_mobile_web foothill lake_lanier contract_t-mobile paris phone_instruction	roomie lip flavor_toothpaste marc tongue 2012-04-25 strip_sample crest_whitening ulta_purchase lecture germ butter amber shield pro_expert sperm crest_kids_sparkle	metal pilot pavement stadium blanket leicester_holiday_inn prob_holiday_inn dance canuck pool_deck polynesian estate_license
	2012-04-26 dispute cellular calling_card moblie cricket-phone textgram	confession intention doctor_dental stank deodorant scope_whiting_mouth_wash issue toothpaste_ad ache braun_electric_toothbrush	2012-04-24 pet robe slipper umbrella budget holiday_inn_bandwidth shite_internet nozzle
		2012-04-26 travel_size laurel bright scope_mouth_wash copenhagen design bottom mojo crest_whitening_advanced_toothpaste	2012-04-25 holiday_inn_priority_club wiz scheme kidnapping cameron weed wiz_khalifa celebrity_news_excitement celebrity shout marijuana_news marijuananews jail sister rapper nerd
			2012-04-26 mouse snoop metro_police hairdryer nashville nashville_holiday_inn priority_club_card clark
Anomalous Events	2012-04-21 android_smartphone price pcd_venture	2012-04-21 fashionista spring_color_trend	2012-04-21 breakfast motel
	2012-04-22 iphone	2012-04-22	2012-04-22 motel hotel
	2012-04-23 iphone android_smartphone price pcd_venture virgin_mobile	2012-04-23 competition	2012-04-23
	2012-04-24 android_smartphone price pcd_venture virgin_mobile text	2012-04-24 competition enamel monger rembrandt_teeth_whitening_kit powerbrush teeth	2012-04-24 holiday_inn_express molina doctor
	2012-04-25	2012-04-25 teeth powerbrush competition crest_pro_health advert	2012-04-25 motel holiday_inn_express molina hotel doctor wiz_khalifa weed wiz
	2012-04-26	2012-04-26 advert crest_pro_health	2012-04-26 wiz_khalifa weed wiz chillin

Figure 4.7: Detected events for three topics.

detection are different from the ones by the streaming one because the streaming event detection would only consider the data in the history, while batched one would consider the data "in the future".

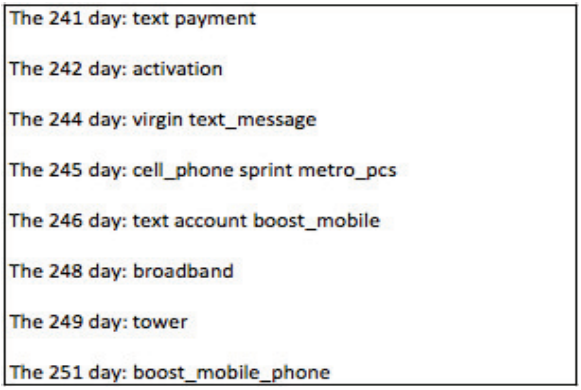


Figure 4.8: Batched event detection results for Sprint-Mobile.

4.8 Experiments for Multi-Task Multi-Label Classification

In this section, instead of classifying tweets from all domains in the social media, we conduct experiments over a single domain — "mobile customer care" to better show the performance of our MTML method.

4.8.1 The Data Set

The real-world tweets, which are related to "mobile customer care", are crawled from 8/31/2010 to 4/26/2011. Specifically, each tweet contains at least one of the following four keywords: "virginmobile, "VMUcare, "boostmobile, and "boostcare. After removing tweets that are posted by company customer services, 6,496 user-generated tweets are obtained. Professionals are invited to select some representative topics from the domain. 10 topics, including "Care/Support, "Lead/Referral, "Mention, "Promotion, "Review, "Complaint, "Inquiry/ Question, "Compliment, "News, and "Company/Brand are finally picked up. While the sentiment labels contains "Positive, "Negative, and "Neutral as usual.

4.8.2 Ground Truth Labeling

Amazon Mechanical Turk (AMT) is employed to assign topic and sentiment labels to tweets. AMT is a crowdsourcing marketplace, which allows people to collaborate with each other to finish hard tasks. By hard, it usually means "hard for machine" instead of human being. There are two types of users in AMT: requesters and workers. Requesters post Human Intelligence Tasks (HITs) with monetary incentives; while workers can browse HITs and complete them for monetary incentives. Requesters accept or reject the result submitted by workers based on its quality.

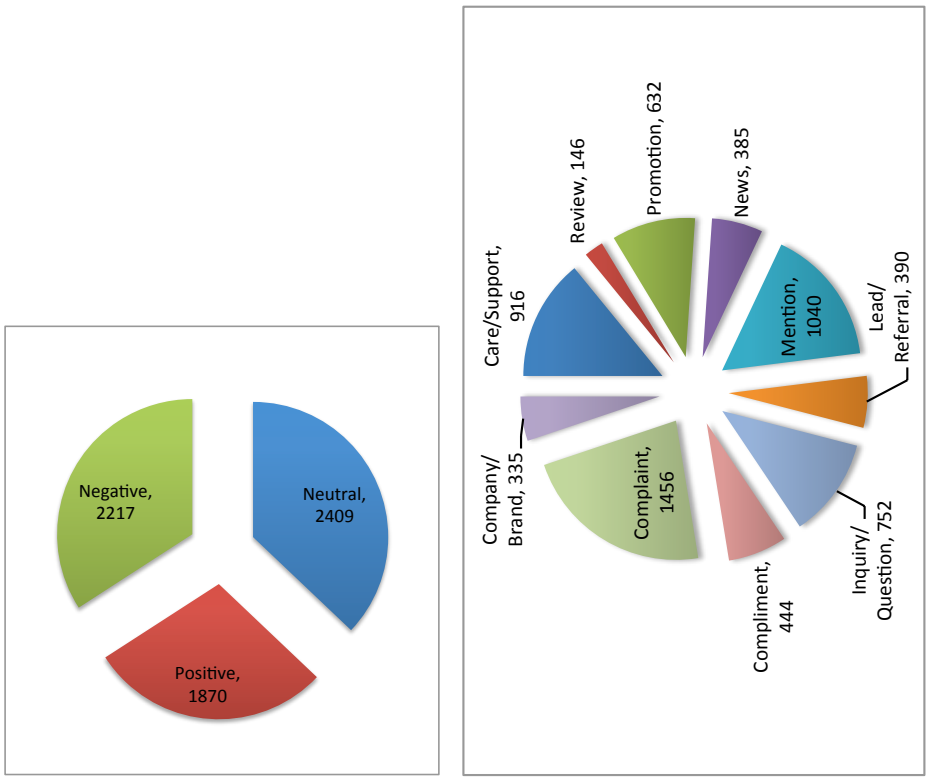
The tweets set was posted in AMT, and three topic labels and three sentiment labels are collected for each tweet via "workers" in AMT. Note: some tweets are "understandable" enough to receive three identical topic/sentiment labels; some others might be labeled with three different topics/sentiments; and the rest are with two identical labels and one different label. In order to obtain "ground truth" labels for each tweet, the identical labels in the first and third cases are picked up due to the "Majority Voting" mechanism; for the second case, the topic/sentiment label is randomly selected as the "true" label.

After the ground truth labeling, the distribution of the tweets on sentiment and topic labels can be shown in Figure 4.9a and 4.9b, respectively.

4.8.3 Feature Selection

As labels of tweets are ready, features of tweets are then generated by extracting keywords from their contents. Hashtags [2] are important keywords in the sense that they represent topics mentioned in tweets. However, they are treated the same as other keywords, without any special weighting or discrimination. Because the "topics" in our case are pre-defined by professionals, the topic information inherent to Hashtags is not necessary.

[2]http://en.wikipedia.org/wiki/Hashtag



(a) (b)

Figure 4.9: (a) Tweets distribution on Sentiment labels; (b) Tweets distribution on Topic labels.

Initially, 50,553 keywords (thus feature dimensions) are extracted. Instead of doing dynamic feature reduction using conventional methods such as PCA, we used a simple empirical approach. We first measured the accuracy while varying the number of features from 400 to 5,000. For the sentiment classification task, the highest ac- curacy was obtained with 3,400 features, while for the topic classification task, 2800 features produce the best result. As a result, in the experiment, we simply adopted the 3,400 and 2,800 features for both sentiment and topic classification tasks, respectively.

4.8.4 Evaluation

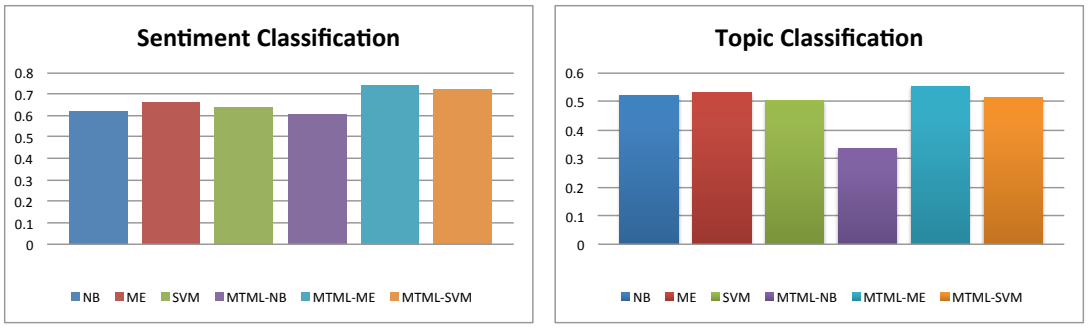
The classification accuracy is used to measure the performance of our method. It is defined as:

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N} I, \tag{4.3}$$

where $I(true) = 1$, and $I(false) = 0$.

The MTML solution is evaluated from the following two aspects: (1) The accuracy of the sentiment classification is computed from the testing results of MTML solution, and then compared against the one of Naive Bayes, SVM, or Maximum Entropy only. (2) The accuracy of the topic classification is computed from the testing results of MTML solution, and then compared against the one of Naive Bayes (NB), SVM, or Maximum Entropy (ME) only.

The comparison results are listed below in Figure 4.10a and 4.10b:



(a) (b)

Figure 4.10: (a) The comparison of sentiment classification. (b) The comparison of topic classification.

One can observe that the MTML solution, which combines sentiment classification and topic classification together, improves the classification performance over these two tasks. Considering its promising performance, it provides an alternative way to summarize the time-sensitive contents of the social media by assigning pre-defined topics to those contents.

4.9 Conclusion

In this chapter, three different methods are proposed to summarize the Time-Sensitive OSN contents, including (1) a novel multi-document summarization

method is proposed to summarize tweets based on events; (2) a event detection framework, which assists audience to watch new and "anomalous" event; and (3) a multi-task multi-label classification to help summarize coming tweets into different topics and sentiments.

4.9.1 Storyline Generation

This is a pioneer work on generating storylines from social media. Different from the traditional multi-document summarization work, it provides audience an alternative way — Storyline to quickly digest the summary of events, news, updates, and trends in the social network.

4.9.2 Event Detection

A first attempt of designing a versatile event detection framework based on the microblog documents for the social media audience is made for preventing the issues raised in traditional event detection methods and assisting them in monitoring the events in the microblog systems in a timely fashion.

4.9.3 Multi-Task Multi-Label Classification

A Multi-Task Multi-Label classification method is explored to shed light on an alternative way for summarizing the time-sensitive contents from the social media into different topics and sentiments.

CHAPTER 5

IDENTIFYING INFLUENTIAL USERS

Identifying influential users and predicting their "network impact" in social networks is an interesting problem in both academia and industry. Various definitions of "influence" and many methods for calculating influence scores have been provided for different empirical purposes and they often lack the in-depth analysis of the "characteristics" of the output influence. In addition, most of the developed algorithms and tools are mainly dependent on the static network structure instead of the dynamic diffusion process over the network, and are thus essentially based on descriptive models instead of predictive models. Consequently, very few existing works consider the dynamic propagation of influence in continuous time due to infinite steps for simulation. In this chapter, we provide an evaluation framework to systematically measure the "characteristics" of the influence from the following three dimensions: i). *Monomorphism vs. Polymorphism*; ii). *High Latency vs. Low Latency*; and iii). *Information Inventor vs. Information Spreader*. We propose a dynamic information propagation model based on *Continuous-Time Markov Process* to predict the influence dynamics of social network users, where the nodes in the propagation sequences are the users, and the edges connect users who refer to the same *topic* contiguously on time. Finally we present a comprehensive empirical study on a large-scale twitter data set to compare the influence metrics within our proposed evaluation framework. Experimental results validate our ideas and demonstrate the prediction performance of our proposed algorithms.

5.1 Overview

5.1.1 Identifying Influential Users

Social network analysis has been gaining attention from different domains, including economics, anthropology, biology, social psychology, physics, etc..

The rapid growth of the online social network sites (e.g. Facebook, Twitter, LinkedIn, and Google+) and their publicly available data acquiring API has led the prosperity of social network analysis research these days. One of most popular topics of the social network analysis is identifying influential users and their "network impact". Knowing the influence of users and being able to predict it can be leveraged for many applications. The most famous application to researchers and marketers is viral marketing [DR01, KKT03, RD02], which aims at targeting a group of influential users to maximize the marketing campaign ROI (Return of Investment). Other interesting applications include search [AA05], expertise/tweets recommendation [STLS06, DYB+07, CNN+10], trust/information propagation [GGLNT04, GH06], and customer handling prioritization in social customer relationship management.

5.1.2 Limitations of Current Research Efforts

There are two main limitations of current research efforts on identifying influential users in social network analysis: one is on the characteristics of influence, and the other is on the influence models and measures.

Characteristics of Influence

Various definitions of "influence" and many methods for calculating influence scores have been provided for their own empirical purposes, or applications. Since they often lack the in-depth analysis of the "characteristics" of the output influence. it is difficult to adapt or choose them for other applications.

Influence Models and Measures

Currently most applications and tools compute user influence based on their static network properties, such as, the number of friends/followers in the social

graph, the number of posted tweets/received retweets/mentions/replies in the activity graph, or users' centrality (e.g. PageRank, Betweeness-centrality, etc.).

A few works investigate adoption behaviors of social network users as the dynamic influence propagation [1] or diffusion process [Rog03]. The adoption behaviors refer to some activities or topics (tweets, products, Hashtags, URLs, etc.) shared among users implicitly and explicitly such as users forwarding a message to their friends, recommending a product to others, joining some groups with the similar musical favor, and posting messages about the same topics, etc. According to the diffusion theory, the information cascades from social leaders to followers. In most diffusion models, propagators have certain probabilities to influence their receivers, and the receivers also have certain thresholds to be influenced. Finding the social leaders or the users who can maximize the influence coverage in the network is the major goal of most diffusion models.

Some drawbacks of existing social network influence models based on either static networks or the "influence maximization" diffusion process are: (1) The static influence scores are not actionable for users. For example, marketers do not know what will be the difference if targeting users with influence scores of 30 or 80. (2) Most existing models are descriptive models rather than predictive models. For example, the number of friends or the centrality score of a given user describes his/her underlying network connectivity. The number of tweets that a user posted or get retweeted indicates the trust/insterest that his/her followers have on his/her tweets. All these measures/models are descriptive and very few models are able to predict users' future influence. (3) Existing "influence maximization" diffusion process is often modeled by discrete-time models such as Independent Cascade Model or Linear Threshold Model. Because the real world diffusion process is continuous-time, it is difficult to define an appropriate time step t for discrete-time models.

[1] In this chapter, we use "information/influence propagation", "information/influence diffusion", and "information cascade", interchangeably to represent the same concept.

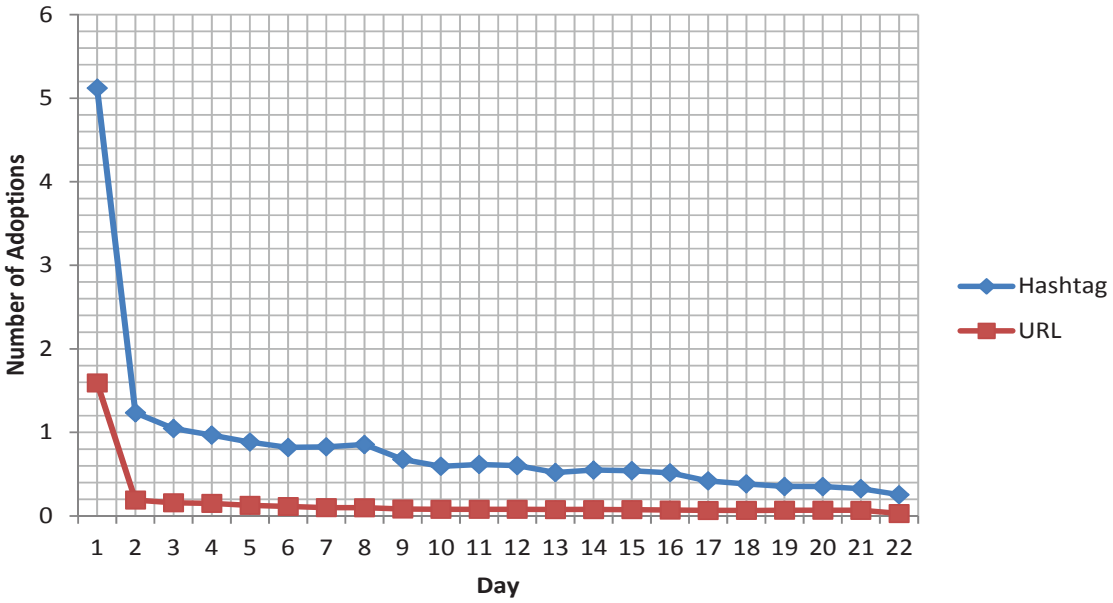


Figure 5.1: The average number of topic adoptions over the time on our Twitter data set.

5.1.3 Content of The Chapter

The aforesaid limitations motivate our study on social network user influence and dynamics prediction in this chapter. In particular, to address the first limitation, we take an initial step to introduce three dimensions of influence: i). *Monomorphism vs. Polymorphism*; ii). *High Latency vs. Low Latency*; and iii). *Information Inventor vs. Information Spreader*, for understanding the characteristics of influential users calculated from various methods. These three dimensions provide an evaluation framework to systematically measure the influence.

To address the second limitation, we propose a dynamic information diffusion model based on the *Continuous-Time Markov Process* (CTMP) to predict the influence dynamics of social network users. CTMP assumes that the number of activations from a given node is following an exponential distribution over the time. This can be often seen in the real-world data [KLPM10]. Figure 5.1 shows that the average number of topic adoptions decreases exponentially over the time. Hashtags receive more adoptions compared with URLs, and the number of Hashtag adoptions decreases more slowly. Furthermore, transition rates q are

calculated and treated as the transition probabilities (or activation probability) of the embedded Markov chain in CTMP. Then the transition probability $P(t)$ can be computed from q, given any time t. In this chapter, the nodes in the propagation sequences are the users, and the edges connect users who refer to the same topic contiguously on time. Topics here particularly refer to Hashtags (expressed as # followed by a word) and short URLs (e.g. bit.ly, TinyURL, etc.) on twitter, which is one of the most popular microblog services, was launched since July 13, 2006. Hashtags and URLs are both unique identifiers tagging distinct tweets with certain "topic" labels. We regard the temporal sequences of Hashtags and URLs as the diffusion paths, where the topics are reposted subsequently. Although retweeting is not included in this chapter as a diffusion approach, it is implicitly considered because the retweets would usually contain the same Hashtags and URLs as in the original tweets. Our experimental results on a large-scale twitter data set show that our proposed diffusion model outperforms other influence models for viral marketing. It also demonstrates a promising prediction performance on estimating the number of influenced users within a given time.

5.1.4 Chapter Contribution and Organization

A preliminary study of the work has appeared at the 15th Asia-Pacific Web Conference in 2013 [LPLS13]. In that conference paper, the study focuses on the proposed influence model – IDM-CTMP, and shows its advantages over two baselines, which are not necessarily continuous-time models. In this dissertation chapter, (1) we propose three "dimensions" of users' influence in the social network to help others understand different aspects of influence; (2) we conducted comprehensive experiment to systematically measure users' influence and compare different influence models over three proposed dimensions; (3) two heuristic continuous-time influence models are defined as baselines to further show the advantages of our proposed model.

In summary, the contributions of this chapter are listed below.

1. We introduce three dimensions on application perspectives and provide an evaluation framework to systematically measure the influence and compare different influence models (See Section 5.6.3).
2. Comprehensive experiments are conducted on various extracted networks (mentions, retweets, replies), as well as temporal propagation paths from the large-scale twitter data (See Section 5.6).
3. Two heuristic influence models considering the topic diffusion in continuous time are defined as baselines (See Section 5.4) to highlight the strengths of our proposed dynamic information diffusion model based on the *Continuous-Time Markov Process*.

The remainder of this chapter is organized as follows. Related work on influence modeling is reviewed in Section 5.2. Before discussing about any influence models, we propose three dimensions of social influence in Section 5.3. After, in Section 5.4, we first give the definition of the temporal influence network, introduce some existing influence models, and propose two heuristic dynamic influence models. In Section 5.5, we propose an information diffusion model based on the *Continuous-Time Markov Process*. Experimental results are demonstrated in Section 5.6. In particular, we discuss the three dimensions of influence and present a comprehensive empirical study on a large-scale twitter data set to compare the influence metrics (including both the dynamic influence metrics and well-known static influence metrics) within our proposed evaluation framework in Section 5.6.3. We evaluate the prediction power of our proposed information diffusion model in Section 5.6.4. Finally Section 5.7 concludes the chapter.

5.2 Related Work

A number of recent works have addressed the matter of user influence on social network. Many of them regard user influence as their network metrics. Kwak et

al. [KLPM10] found the difference between three influence measures: number of followers, page-rank, and number of retweets. Cha et al. [CHBG10] also compared these three measures, and discovered that the number of retweets and the number of mentions are correlated well with each other while the number of friends does not correlated well with the other two measures. Their hypothesis is that the number of followers of user may not be a good influence measure. Weng et al. [WLJH10] regarded the central users of each topic-sensitive subnetwork of the follower-and-followee graph as influential users. Other work such as [GL10, RGAH10, ALTY08, TSWY09] mined users influence from their static network properties derived from either their social graphs or activity graphs.

Various dynamic diffusion models have also been proposed to discover the influential users. They are shown to outperform influence models based on static network metrics [RD02, GL10]. A lot of work in this direction are devoted to *viral marketing*. Domingos and Richardson [DR01, RD02] were the first to mine customer network values for 'influence maximization' for viral marketing in data mining domain. The proposed approach is a probability optimization method with the hill-climbing heuristics. Kemper et al. [KKT03] further showed that a natural greedy strategy can achieve 63% of optimal for two fundamental discrete-time propagation models - Independent Cascade Model (IC) and Linear Threshold Model (LT) . Many diffusion models assume the influence probabilities on the edges or the probability of acceptance on the nodes are given or randomly simulated. Goyal et al. [GBL10] proposed to mine these probabilities by analyzing the past behavior of users. Saito et al. [SKOM10a, SKOM10b] extend IC model and LT model to incorporate asynchronous time delay. Model parameters including activation probabilities and continuous time delay are estimated by Maximum Likelihood. Our proposed diffusion model is different from the above discussed models: (1) We model the dynamic probabilities of edge diffusion and node threshold changing over the time, rather than computing the static probabilities. (2)

Our model is a Continuous-Time diffusion model instead of a discrete-time diffusion model. Although Saito et al. also proposed Continuous-Time models, the fundamental diffusion process of their models are following LT and IC models. For example, in asynchronous IC, an active node can only infect one of its neighbors in one iteration, while our proposed models does not assume iterations so that nodes can be activated at any time without resetting the clock in the new iteration. Moreover, the models proposed by Saito et al. supposed only one initial active user and focused on model parameter estimation, not much on prediction. The experiments are evaluated on simulated data from some real network topology. Our proposed model estimates the model parameters from the real large-scale social network data, allows many initial active users asynchronously or simultaneously to influence other users, and predicts the real diffusion sizes in the future.

In addition, most of influence models are basically descriptive models instead of predictive models. Bakshy et al. [BHMW11] studied the diffusion tree of URLs on twitter, and train a regression tree model on a set of user network attributes, user past influence, and URL content to predict users' future influence. Our work is different from the work of Bakshy et al. in the following aspects.

1. They predict users average spreading size in the next month based on the data from the previous month. However, the dynamic nature of word-of-mouth marketing determines that the influence coverage vary over the time. Thus our work aims at predicting the spreading size of each individual user within a specific given date, so we can answer "what is the spreading size of user A within 2 hours, 1 day, or 1 month, etc.".
2. Their work is based on a regression model. While we proposes a real-time stochastic model. The input and output of these two models are different.
3. Besides URLs diffusion, we also study the diffusion of Hashtags on twitter, which usually have longer lifetime.

Continuous-Time Markov Process (CTMP) has been used in web-page or document browsing. Huang et al [HYHN04] adopted it to model the web user visiting patterns. Liu et al. [LGL+08] also utilized Continuous-Time Markov Process to model user web browsing patterns for ranking web pages. Song et al. [SCHT07] employed CTMP to mine document and movie browsing patterns for recommendation. To the best of our knowledge, our work is the first to construct influence diffusion model based on CTMP for spreading coverage prediction and user influence on social networks. We are also the first to introduce three intuitive criteria for users to compare and choose different influence models.

5.3 Three Dimensions of Influence

Everyone is talking about how to identify influential users, because it is believed that those users can help with many applications, e.g., Viral Marketing. However, what does influence exactly mean in the context of social media?

In this section, social media users' influence is discussed from three dimensions.

5.3.1 Monomorphism VS. Polymorphism

The concept of Monomorphism vs. Polymorphism is borrowed from the diffusion of innovations [Rog03]. In this dissertation, users with high monomorphism usually focus on a constant set of topics, while users with high polymorphism post a variety of topics over the time. Knowing this property of social media users could benefit applications with different purposes. For example, high monomorphism influencers should be ranked higher than high polymorphism influencers in expert recommendation applications. However, the high polymorphism influencers would be more desirable to users aiming for general information gathering.

To determine whether a user is monomorphic or polymorphic is difficult, nevertheless, we suppose if a user is monomorphic, his/her posted topics should be similar across two different time periods; on the other hand, if a user is polymor-

phic, his/her topics would be different across two different time periods. In our experiment, we compare two time periods – the 12-day training period and 10-day testing period specified in previous sections. For each user, two topic vectors (consisted of Hashtags/URLs) from these 12-day data and 10-day data are extracted. Then the cosine similarity is measured between these two topic vectors as the topic similarity. The high topic similarity indicates the high monomorphism.

5.3.2 High Latency VS. Low Latency

As for the second dimension - *High Latency vs. Low Latency*, here latency means, once a user posted a topic, the time delay before the next posts about the same topic would appear. Influencers with a low latency often receive immediate topic "adoption". Thus they should be picked as viral marketing "seeds" when marketers want to quickly test customer response.

Different topics may result in different adoption latencies. For example, influencers interested in "Machine Learning" might generally have a higher latency than ones interested in "Justin Bieber". Instead of regarding the average time difference between the user's original post and the next topic adoption as the latency, which may be highly affected by the type of topic, we define the latency as follows:

$$\begin{aligned} & Latency(v) \\ =&|\{\tau_1|aveDiff(\tau_1) < firstDiff(v,\tau_1), \tau_1 \in T(v)\}| \\ -&|\{\tau_2|aveDiff(\tau_2) >= firstDiff(v,\tau_2), \tau_2 \in T(v)\}|, \end{aligned} \tag{5.1}$$

where $T(v)$ denotes all the topics posted by user v, τ_1 and τ_2 represent topics from $T(v)$, $aveDiff(\tau_1)$ is the average interarrival time between every pair of neighboring posts about topic τ_1, and $firstDiff(v,\tau_1)$ indicates the time taken for the follower to make the first adoption right after v posts topic τ_1. A large value of $Latency(v)$ means a high latency.

5.3.3 Information Inventor VS. Information Spreader

The third dimension about *Information Inventor vs. Information Spreader*, is to measure the diffusion power of influencers. Information inventors are innovators who are usually the information source, the first group of people to adopt products/brands, or new trend leaders. Information Spreaders are people who are able to spread topics to a lot of social media users. It is quite obvious that the third property dimension of influential users is very useful for viral marketing. The targeted seed users for viral marketing should be both information inventors and information spreaders.

Rather than identifying who are the information inventors and information spreaders, we measure the inventing ability of each user as:

$$Inv(v) = \frac{\#(new\ topics\ started\ by\ v)}{\#(tweets\ by\ v)}. \tag{5.2}$$

The term "new topics" indicates the Hashtags/URLs that are first posted in Twitter. The spreading ability can be computed by using the definition of Time-Window Diffusion Size in Section 5.4.2.

5.4 Influence Network and Influence Models

5.4.1 Influence Network

A social graph can be denoted as $G(V, E)$, where V represents social network users, and E is the set of edges/relations between users. The follower-followee graph is one type of social graphs, where the edges indicate following relations. *Activity graphs* are another type of social graphs, which are extracted from users tweeting behaviors. The typical twitter activity graphs are tweet-retweet graph, tweet-reply graph, and mention-mentioned graph. In this chapter, we run well-known user influence models (e.g., degree-centrality, PageRank) on these three activity graphs in our comparative study. Both the follower-followee graph and activity graphs are directional *Influence Networks*, where the influence flows from

users to people who follow them, or people who retweet their tweets, or people who reply their tweets, or people who mention their names. The influence network can be denoted as $G(V, E_{influence})$, where V denotes social network users, and the edge $V_i \leftarrow V_j$ in $E_{influence}$ means V_j is influenced by V_i.

The above networks can be viewed as static networks, which do not demonstrate the dynamic propagation process over the time. In order to analyze how topics are passing on social networks progressively, we construct a temporal influence network by considering the continuous time. Given a Hashtag/URL (*topic*), a group of users can be ordered based on the time when they post this topic. As shown in Figure 5.2, user i is linked to user j if they post the same topic contiguously and user j follows/friend with user i. The number on the top of each arrow is the time taken to transfer a topic from a user to another user.

Definition 5.4.1 (Temporal Influence Network) *The temporal influence network is* $G(V, E, T(E))$*, where* $V = \{V_0, V_1, \cdots, V_n\}$ *contains all users who posted at least one Hashtag or URL,* $E = \{V_i \leftarrow V_j | V_i$ *posted a topic earlier than* $V_j\}$*, where edges can be constrained to only exist between followers and followees or between friends. So the propagation is along the paths from followees to followers over continuous time. The function* $T(V_i \leftarrow V_j) = \{t_{ij}^0, t_{ij}^1, \cdots, t_{ij}^l\}$*.* $t_{ij}^m \in \{t_{ij}^0, t_{ij}^1, \cdots, t_{ij}^l\}$ *is the time difference between user i posting a topic and user j posting the same topic.*

There can be multiple entries in $T(V_i \leftarrow V_j)$ since user i and user j can post the same set of topics or one topic at multiple times. Note that we aggregate all topics together to form this temporal influencer network in this chapter. One natural extension is to categorize these Hashtags/URLs into topics so that topic-sensitive influential users can be computed from each topic-dependent network G_{topic_i}.

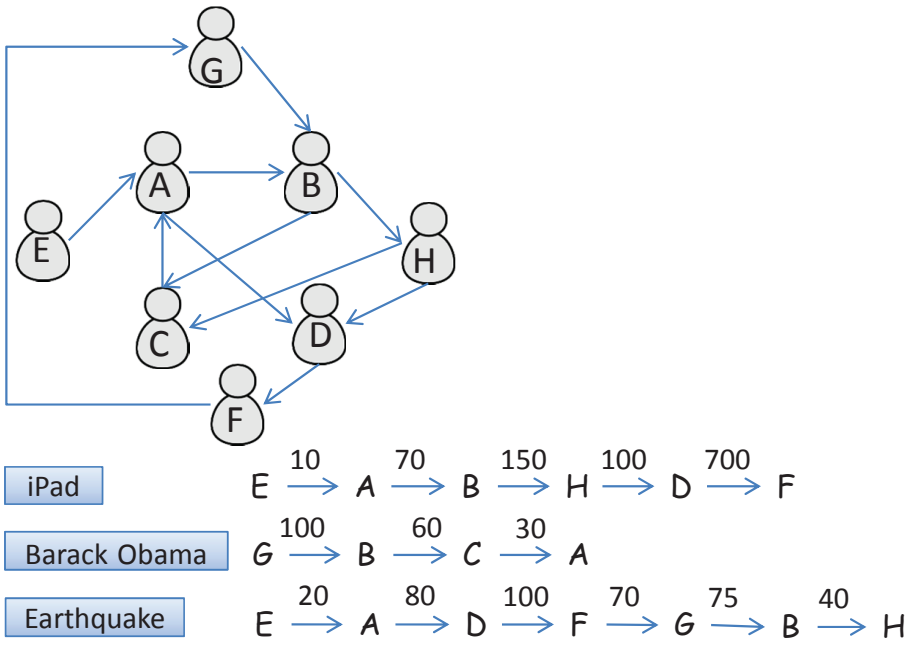


Figure 5.2: The example of Temporal Influence Network construction.

5.4.2 Influence Models

Degree Centrality and PageRank, as two most well-accepted influence models, are computed on static networks. The static networks here refer to the three activity networks we specified in previous subsection. The Degree Centrality is defined as the number of inlinks incident upon a node/vertex. The essential idea of PageRank is to define a link analysis method to evaluate a user's influence, so that not only the immediate information flow is incorporated, but also the information flow after that would be considered. According to PageRank, a user is "authoritative" if he/she has a lot of inlinks from other "authoritative" users.

Different from the above mentioned influencer models, we propose two straightforward dynamic influence models on the temporal influence network to incorporate the temporal information.

Time-Window Diffusion Size

Definition 5.4.2 (Time-Window Diffusion Size) *The diffusion size of a user u over a topic c, $DS_{u,c}$, is the number of other users posting the same topic c after*

user u within a pre-defined time range. The aggregated diffusion size over all the topics of a user is $DS_u = \sum_c DS_{u,c}$.

It is worth noticing that the influence computed here is based on a pre-defined time range, specifically, this method grants us the ability of identifying the comparative influential users within a pre-defined time range. We can see that the users with a large time-window diffusion size tend to post topics at the beginning of fast and large cascade of the topics.

Temporal Closeness Centrality

Definition 5.4.3 (temporal distance) *The temporal distance* $d_{temporal}(V_i, V_j)$ *between two users* V_i *and* V_j *is the least time difference* $min(T(V_i \leftarrow V_j))$ *w.r.t. the set of topics posted by both* V_i *and* V_j *where* $T(V_i \leftarrow V_j)$ *is defined in Definition 5.4.1.*

In order to measure the *reach-ability* of a user, the temporal closeness centrality is given by:

$$TCC_u = \frac{\sum_{v \in V \setminus u} d_{temporal}(u, v)}{n - 1}, \tag{5.3}$$

where n is the number of all users in the temporal influence network. It is worth pointing out that: sometimes a user u never goes to v since no topic diffuses from user u to v. In such a case, we treat the temporal distance between u and v as $n \cdot Max_{i,j \in V, i \neq j} T\{V_i \leftarrow V_j\}$. Users with low temporal closeness centrality often post topics close to fast and large cascade of the topics.

5.5 Information Diffusion Model based on Continuous-Time Markov Process

The aforementioned influence models are either based on static activity networks or descriptive models (instead of predictive models) building on the temporal influence network. The descriptive models answer questions such as "How many

followers that user A has?" and "How many followers post the topic 'ipad' after user A?", etc. In this section, we introduce our proposed predictive Information Diffusion Model based on Continuous-Time Markov Process, abbreviated as IDM-CTMP for convenience. IDM-CTMP is able to answer the following question, "In the next month, how many users would post the topic 'ipad' estimably if user A posts it now.", or even a harder question "In order to make a maximal number of people to talk about our product in the next week, who are the seed users we should target?". Note the influential users discovered by IDM-CTMP maximize not only the information *coverage*, but also the *rate* of information cascade given a certain period of time.

5.5.1 Model Formulation

A trending topic (a Hashtag/URL) is propagated by social network users within the temporal influence network defined in Definition 5.4.1. Suppose $X(t)$ denotes the user who posts a specific topic at time point t, $X = X(t), t \geq 0$ forms a Continuous-Time Markov Process (CTMP) [AJ91], in which the user who will discuss this topic next only depends on the current user given the whole history of the topic propagation. Formally, this markov property can be defined by:

$$\begin{aligned} P_{ij}(t) &= P\{X(t+\gamma) = j | X(\gamma) = i, X(\mu) = x(\mu), 0 \leq \mu < \gamma\} \\ &= P\{X(t+\gamma) = j | X(\gamma) = i\}, \end{aligned} \tag{5.4}$$

where $P_{ij}(t)$ is the transition probability from i to j within time t, i is the current user who discusses the trending topic, j is the next user who posts the topic following i, and $x(\mu)$ denotes the history of the topic propagation before the time point γ. We assume that the transition probability $P_{ij}(t)$ does not depend on the actual starting time of the propagation process, thus the CTMP is time-homogeneous:

$$\begin{aligned} P_{ij}(t) &= P\{X(t+\gamma) = j | X(\gamma) = i\} \\ &= P\{X(t) = j | X(0) = i\}. \end{aligned} \tag{5.5}$$

In order to estimate the diffusion size of user i given a pre-defined time window t, we need to compute the transition probability from user i to all the other users, then determine the number of users being affected by i at the end of the time window. The diffusion size of user i over time t based on CTMP can be defined as

$$DS_{i,t} = \sum_{j} P_{ij}(t) \cdot n_i, \tag{5.6}$$

where n_i is the number of times that user i occurs at time t. It can be estimated by supposing that it linearly increments on t. However, it is impractical to estimate the transition probability matrix $P(t)$ with infinite possible t. Thus instead of estimating $P(t)$ directly, we calculate the transition rate matrix Q, and then $P(t)$ can be estimated from Q.

5.5.2 Estimation of Transition Rate Matrix

The transition rate matrix Q is also called the infinitesimal generator of the Continuous-Time Markov Process [Dyn65]. It is defined as the derivative of $P(t)$ when t goes to 0. The entry q_{ij} is the transition rate to propagate a topic from user i to user j. The sum of the rows in Q is zero, with $\sum_{j,j\neq i} q_{ij} = -q_{ii}$.

$$q_{ij} = \lim_{t\to 0} \frac{P_{ij}(t)}{t} = P'_{ij}(0) \quad (i \neq j). \tag{5.7}$$

Note that q_{ij} reflects a change in the transition probability from user i to user j. q_i, namely out-user transition rate in this chapter, is equal to $-q_{ii}$. It indicates the rate of user i propagating topics to any other users.

Figure 5.1 presents the average number of topic adoptions in each day of the 22 days. One can conclude from this figure that the average number of topic adoptions decreases exponentially over the time. Thus, in order to compute q_i, we assume that the time for user i to propagate a topic to all the others is following an exponential distribution as observed for many users in our data, where the rate parameter is q_i.

The expected value of an exponentially distributed random variable T_i (in this case, the topic propagation time for user i) with rate parameter q_i is given by [Fel08]:

$$E[T_i] = \frac{1}{q_i}. \tag{5.8}$$

Thus q_i is one divided by the mean of $\cup_j(T(V_i \leftarrow V_j))$, which is defined in the temporal influence network.

According to the theory of Continuous-Time Markov Process, if a propagation occurs on user i, the probability that the other user j would post the topic forms an embedded Markov chain [KT75]. The transition probability is S_{ij}, and $\sum_j S_{ij} = 1$ $(i \neq j)$ and $S_{ii} = 0$. One important property is that $q_{ij} = q_i S_{ij}$. Then, the transition rate from user i to j can be estimated by:

$$q_{ij} = \sum_m q_i^2 \cdot exp(-q_i \cdot t_{ij}^m), \tag{5.9}$$

where m is the number of topics diffused from use i to j, and t_{ij}^m denotes the transition time from user i to j on the m-th topic.

5.5.3 Estimation of Transition Probability Matrix

Now we obtain all the entries of the transition rate matrix Q. Next, we will specify how to derive the transition probability matrix $P(t)$. The well accepted Kolmogorov's Backward Equations [Gar85] in the Continuous-Time Markov Process can be utilized:

$$P'_{ij}(t) = q_i \times \sum_{i \neq k} P_{ik}(t) \times P_{kj}(t) - q_i \times P_{ij}(t). \tag{5.10}$$

By performing some algebraic operations, the above equation can be written as the following matrix form:

$$P'(t) = QP(t). \tag{5.11}$$

The general solution for this equation is given by:

$$P(t) = e^{Qt}. \tag{5.12}$$

$P(t)$ is a stochastic, irreducible matrix for any time t. We approximate it using Taylor expansion, so that $P(t)$ can be estimated by [HYHN04]:

$$P(t) = e^{Qt} = \lim_{n \to \infty} (I + Qt/n)^n. \tag{5.13}$$

We raise the power of $(I + Qt/n)$ to a sufficiently large n.

5.6 Experiment

5.6.1 The Data Set Description

Twitter provides Streaming APIs which allow high-throughput near-realtime access to various subsets of twitter data. It samples the statuses (including the tweets and the authors) from the Firehose stream of public statuses which is the full feed of all public tweets. This dissertation uses Twitter Gardenhose streaming API, which is said to sample 10% of all public tweets. Hashtags beginning with # in tweets represent keywords or topics. URLs add more detailed topic information to tweets, shortened via the services such as bit.ly or tinyurl.com. Hashtags and URLs enable twitter users to create and follow a thread of discussion. They are regarded as unique identifiable topics in this dissertation. Hashtags and URLs of each tweet can be extracted from its metadata fields, embedded in the crawled raw twitter data.

We continuously collected 22-day twitter data, ranging from March 2 to March 24, 2011. The first 12-day data is used for our training purpose, and the remaining 10-day data is for testing and validation. We removed all non-English tweets to focus on only English twitter world. Tweets posted by users with less than 20 followers are also removed. These twitter users post close to 10% of all tweets, and supposedly they are very less likely to be influential users. Finally we have a total of 48,113,490 tweet records (a tweet record may include the tweet and the corresponding retweets, thus more than one tweet) in the 12-day training data. As for the 10-day test data, we also removed tweets without Hashtags or URLs, which

result in 27,237,631 tweet records, about 10% of original data. We also filter the tweets with more than three Hashtags or URLs, which tend to be spam tweets as introduced in [KLPM10]. Finally, we obtained totally 78,858,046 tweets, in which there are 9,431,404 unique users, 3,209,330 Hashtags, and 21,107,164 URLs.

5.6.2 Correlations Between Different Metrics

In order to understand the difference between various influence metrics: the number of mentions, the number of replies, the number of retweets, the PageRank of mentions, the PageRank of replies, the PageRank of retweets, Time Window Diffusion Size, Temporal Closeness, and IDM-CTMP, we measured the overlap and Spearman's correlation between every two influential user rank lists obtained from the above metrics. Although our proposed IDM-CTMP is a dynamic metric which outputs different user rank lists given different time ranges, in our comparative study we fix the time range to be 10 days (from day 13 to day 22). The social graph is not added as a constraint on IDM-CTMP in the experiment to test its performance even without any network structure information. The empirical result shows little correlation between most pairs of rank lists except the correlation between the number/PageRank of mentions and the number/PageRank of replies, and the correlation between the Time Window Diffusion Size and Temporal Closeness.

5.6.3 An Evaluation Framework to Measure Three Dimensions of Influence

To understand the properties of various user influence metrics, we conducted experiments to systematically compare them on three dimensions: i). *Monomorphism vs. Polymorphism*; ii). *High Latency vs. Low Latency*; and iii). *Information Inventor vs. Information Spreader*.

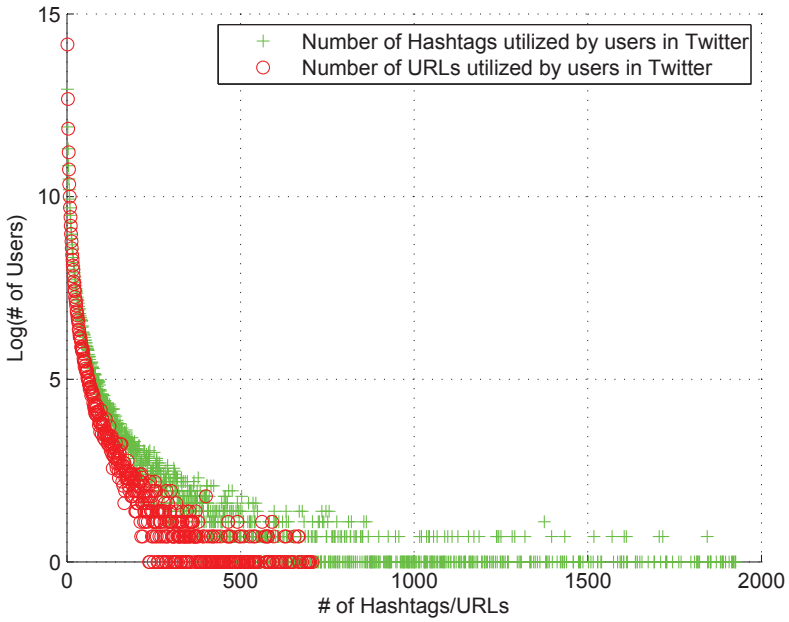


Figure 5.3: Number of Hashtags/URLs utilized by users in Twitter.

Monomorphism VS. Polymorphism

Figure 5.3 shows that a large number of users have ever used a limited number of Hashtags/URLs, while only a few users utilized a large quantity of Hashtags/URLs. That means a very few twitter users post a wide range of topics.

As we introduced in Section 5.3.1, the cosine similarity of two topic vectors from the first 12-day training period and 10-day testing period for a specific user is used as the topic similarity. The high topic similarity indicates that this user has high monomorphism.

To compare 9 different user influence rank lists, we choose the top 10,000 users and the bottom 10,000 users from each rank list. The average topic similarity of the top 10,000 and the bottom 10,000 users for each rank list across the specified two time periods is computed.

The comparison results are presented in Figure 5.4. From the results, we can observe that users with high degree centralities of mentions and retweets have higher monomorphism. Especially the gap between the top 10,000 and the bottom 10,000 users based on the number of mentions is the largest. Looking into the

data, users with high mentions like justinbieber, charliesheen, JonasBrothers, and XSTROLOGY, usually focus on a constant set of topics. Note that the top ranked users by our proposed method IDM-CTMP have relatively high polymorphism. Thus they tend to post a variety of topics over the time. In order to explain this phenomenon, we can think of the characteristics of the data set and the IDM-CTMP method. The data set is sampled from the real world tweets, which contain all kinds of topics, thus it covers topics from different areas. IDM-CTMP aims at identifying influential users who are able to diffuse topics to many other users no matter from which areas those topics come. Therefore, polymorphic users tend to be ranked higher by IDM-CTMP since they can diffuse more topics to more users in the social network.

High Latency VS. Low Latency

To measure the latency of influential users, we first calculate the latency score as shown in Equation 5.1 for each user. Then 10,000 users with the lowest latency are compared with the top 10,000 users from each user influence rank list using the Spearman's correlation. The correlation results are shown in Figure 5.5. It shows that the top ranked influential users from IDM-CTMP have the lowest latency than other metrics. The reason behind this observation is that IDM-CTMP tries to maximize not only the diffusion coverage but also the diffusion speed.

Information Inventor VS. Information Spreader

Similar to previous experiments, 10,000 users with the highest information inventing ability score and 10,000 users with the highest information spreading ability score are extracted to compare with the top 10,000 users from 10 influential user rank lists using the Spearman's correlation.

The comparison results are shown in Figure 5.6. In this experiment we add "INV", which is the metric defined by Equation 5.2. Note all the user influence

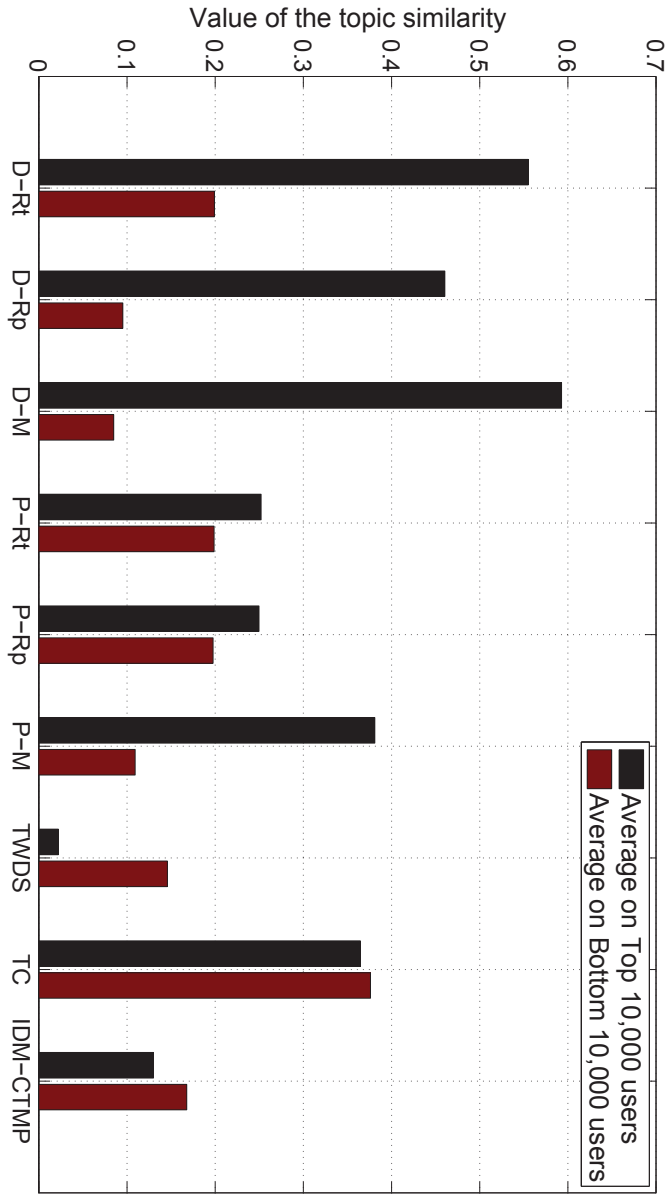


Figure 5.4: The average topic similarity of top 10,000 users and bottom 10,000 users from 9 user influence rank lists. D denotes Degree, P denotes Pagerank, Rt denotes Retweet, Rp denotes Reply, M denotes Mention, TWDS is Time-Window Diffusion Size, and TC means Temporal Closeness.

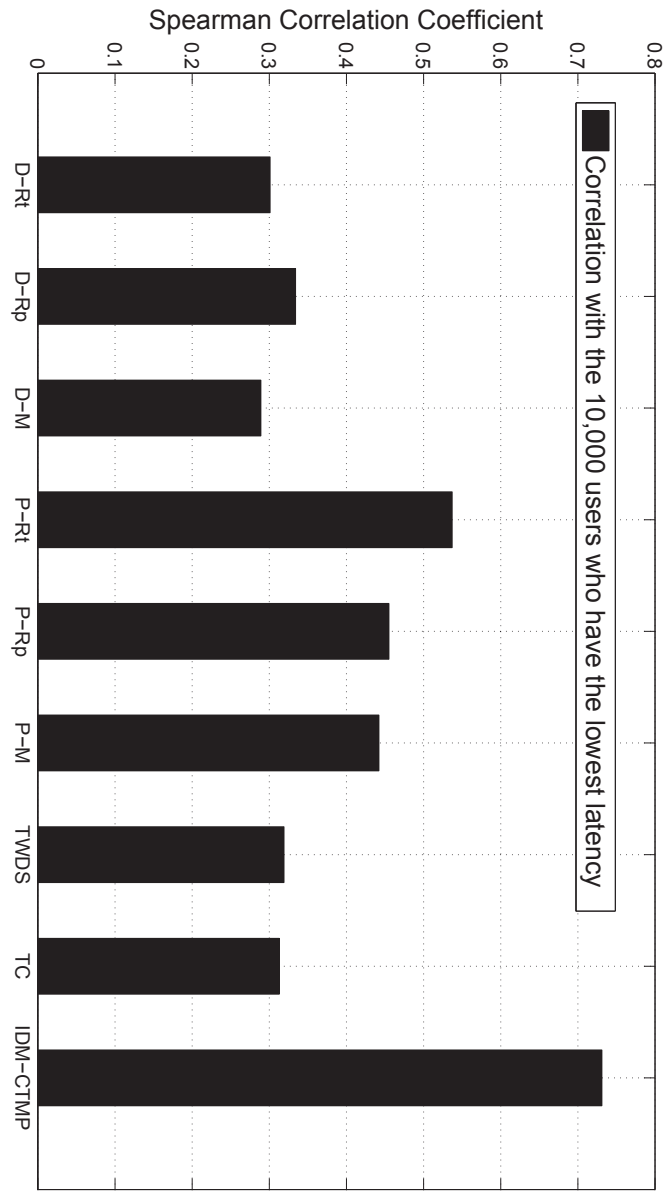


Figure 5.5: The correlation between top ranked 10,000 influential users based on different influence metrics and 10,000 users with the lowest latency. D denotes Degree, P denotes Pagerank, Rt denotes Retweet, Rp denotes Reply, M denotes Mention, TWDS is Time-Window Diffusion Size, and TC means Temporal Closeness.

metrics are calculated from the first 12-day training data, while the ground truth is computed from 10-day test data. Thus the correlation is not 1 for "INV" and "TWDS" compared with the top 10,000 inventing ability users and the top 10,000 spreading ability users, respectively.

We can observe that influential users from all the traditional static metrics do not have high inventing ability and spreading ability. As we expected, "Inv" influencers have high inventing ability but low spreading ability. On the other hand, "TWDS" influencers have high spreading ability but low inventing ability. Our proposed method IDM-CTMP achieves both high inventing ability and high spreading ability because (1) as we described in section 5.6.3, IDM-CTMP can identify high polymorphic users who post many different topics, it is not difficult to conclude that part of those topics are "invented" by those users with high inventing capability; (2) IDM-CTMP focuses on users who can diffuse topics to many other users, in other words, high spreading ability.

It is worth noticing that so far IDM-CTMP has shown its advantages for viral marketing application because its derived top ranked influential users tend to be innovators, obtain quick topic adoption, and spread topics widely and fast.

Comparisons of Top Ranked Influential Users On Twitter

In this section, we list the top 10 influential users identified by various methods in Table 2. Each column presents 10 top influential users found by its corresponding method. The first three columns on the left are based on existing Degree Centrality influence model over "Retweet", "Reply" and "Mention" activities respectively; the next three columns are based on the PageRank influence model, which is then followed by the two baseline approaches: Time-Window Diffusion Size and Temporal Closeness Centrality. Our approach is listed at the last column named "IDM-CTMP". Several observations can be drawn by comparing and analyzing the top influential user lists in Table 5.1:

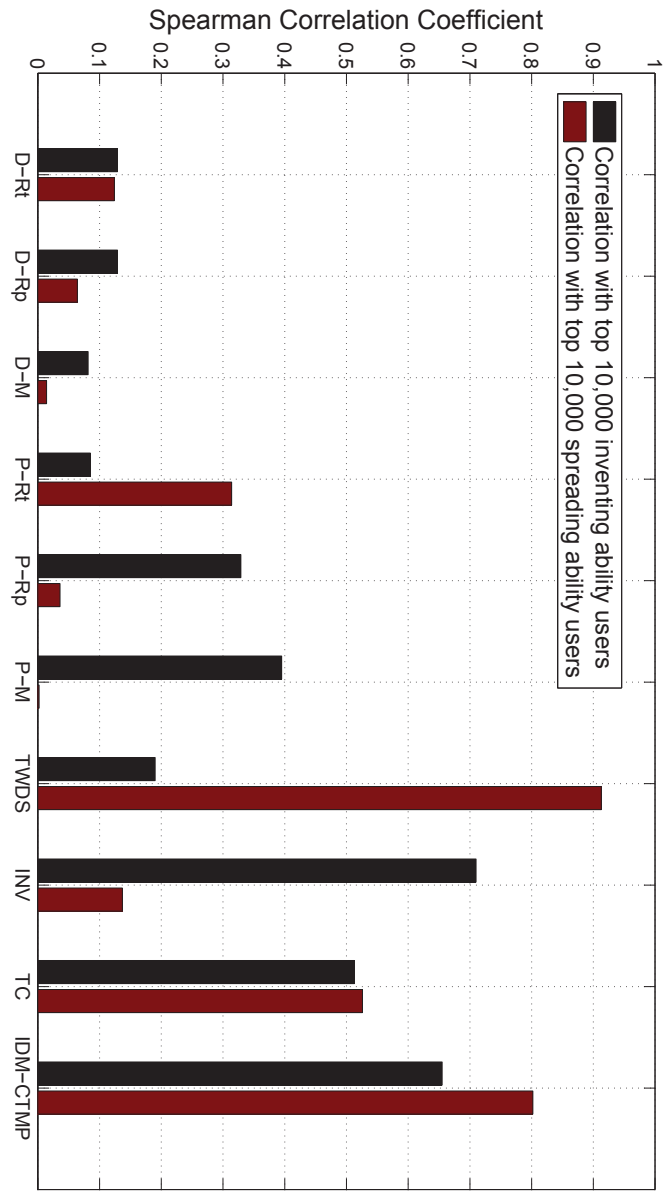


Figure 5.6: The comparison results of top 10,000 users from 10 influence rank lists against top 10,000 inventing ability users and top 10,000 spreading ability users. Notice that D denotes Degree, P denotes Pagerank, Rt denotes Retweet, Rp denotes Reply, M denotes Mention, TWDS is Time-Window Diffusion Size, INV is Inventing Ability metric, and TC is Temporal Closeness.

Degree Retweet	Degree Reply	Degree Mention	PageRank Retweet	Pagerank Reply	Pagerank Mention	TimeWindow DiffusionSize	Temporal Closeness	IDM-CTMP
justinbieber	justinbieber	justinbieber	AntonioPires	XboxSupport	106andpark	bananatay	RT_com	washingtonpost
charliesheen	charliesheen	Xstrology	13eatSmith	FART_ROBOT	drdrew	Ciaramedlies	TrendingUSA	GuyKawasaki
YouTube	officialjaden	ZodiacFact	XboxSupport	bot_marley	glennbeck	FrostBight	iMarriedaWhore	nytimes
JonasBrothers	ochocinco	EpicTweets	waze	ro_bot_dylan	simonpegg	JarrodEder	HumorORtruth	BBCWorld
XSTROLOGY	CodySimpson	TheNoteboook	GibbsRules	yodaism	algore	Neoley	SatelliteShow	kidsjoycom
AddThis	chrisbrown	algore	alfian_007	RedScareBot	priyankachopra	naira_24	GIRLTHINGS	Drudge_Report
foursquare	KimKardashian	damnitstrue	FART_ROBOT	waze	MatchupChats	anasathia	iRespectFemales	cnnbrk
chrisbrown	JASMINEVILLEGAS	charliesheen	jojokejohn	saferprint	Nea1968	kimchidiction	cnnbrk	HuffingtonPost
damnitstrue	rioferdy5	ihatequotes	saferprint	IAmJacksBot	Bikertwitts	mrPerd	eienKATTUN	Metro_TV
officialjaden	selenagomez	WowTeenagers	YouTube	for_a_dollar	BestAt	WhiteAddict	takaosaito	mashable

Table 5.1: The top 10 influential users lists obtained by different methods.

1. Degree Centrality influence model over three activity graphs (namely retweet, reply and mention) consistently picks out the "celebrities" (e.g. "justinbieber", "charliesheen"), who tend to have a large number of followers (or fans). Even though these celebrities may not "tweet" often, but even one or two tweets could still drive significant activities (i.e. retweet, reply and mention) among their immediate followers or fans; not to mention the case in which they "tweet" a lot. Meanwhile, in our experiment settings, Degree Centrality model aims to capture those users who are being frequently retweeted, replied or mentioned. As a result, the Degree Centrality model, which focuses on the first hope of influence (between the celebrity and his/her followers), is more suited for finding out influential celebrities.

2. Similar to the original PageRank algorithm for ranking web pages, the influence model based on the PageRank algorithm identifies the top influential users not just from the one-hop influence, but also based on the influential users from whom they receive "influence", and then spread their influence in the network that beyond their direct followers. In the PageRank's "being-retweeted" and "being-replied" influence networks, the most influential users are not necessarily the celebrities with many followers, but the users who are highly interactive and responsive. For instance, "XboxSupport" and "waze" are twitter user accounts that frequently reply/being-replyed and tweet/being-retweets with their brand customers who seek help for their questions or issues. It is not difficult to imagine scenarios in which the proposed solution to certain customer issues can be spread further in the network beyond their direct followers. Another interesting observation in PageRank-Mention column is that two controversial political figures "glennbeck" and "algore" are listed among the top influential users. This could coincide with some heated political debates during our experimental data gathering time period. These political debates tend to debate on pre-known topics (such as conservative or liberal views on environment, same-sax marriage, etc.),

but the influences spread across the network widely rather than among only direct followers.

3. The top influential users identified by the Time-Window Diffusion Size mostly tend to be some regular guy (or "nobody") who has a small set of followers and followees. But those guys may happen to tweet often on a small set of topics during a particular time-window and their posts get spread to many users who are not necessarily their direct followers. It is very difficult to justify that these people are "influential" in a reliable or consistent manner. Please also be noted that none of these influential users overlaps with the results of our method (IDM-CTMP), which indicates that our method is able to properly filter these users out.

4. Our method, IDM-CTMP, has the ability to identify some of Twitter accounts that representing popular news media (e.g. "washingtonpost", "nytimes", "BBCWorld") along with a well-known influential figure in technology innovation and entrepreneurship (e.g. "GuyKawasaki"). Since our experiment data was collected during March of 2011, during which there was an earthquake and tsunami event in Japan, and also coincidently Guy Kawasaki was on the promotional social-media tour for his new book "The Art of Enchantment". These noticed events have triggered some time-sensitive "new" or "unprecendented" or "bursted" topics. On the other hand, the news media also tend to be either "information innovator" (first mention the topic) or "information spreader" (diffuse news more reliably than regular people) or both at the same time. These correlations prove our method is able to detect not only time-sensitive new topics, but also considering both "innovator" and "spreader" factors. Furthermore, there is some overlap among the top influential users between Temporal Closeness and our IDM-CTMP models. Many of top influential users in Temporal Closeness column belong to news media accounts. This overlap further indicates our IDM-CTMP also favors the low latency of influence spreading.

5.6.4 Predicting Spreading Size Using IDM-CTMP

In the previous subsections, we transformed IDM-CTMP into a static influence metric and compare it with other existing static influence metrics. However, IDM-CTMP is a dynamic metric and a predictive model. It is able to predict how many users would adopt some topic given a certain period of time after a user posts it. In our experiment, we first train IDM-CTMP model on the first 12-day training data, then calculate the spreading coverage of each user for each day from day 1 to day 22, including both training and testing periods.

The Ground Truth

In order to evaluate the prediction performance of IDM-CTMP and present its feasibility in real world applications, we need to provide the ground truth. Suppose user u posts a topic τ at time t_1, and subsequently n users post the same topic τ till time t_2, then the ground-truth spreading size of u from t_1 to t_2 with regards to topic τ, denoted as $DS_{u,\tau}^{t_1 \sim t_2}$, is n. For example, we know that user B first posts a topic #ipad in day 2, afterwards there are 10 users posting #ipad in day 2, and 20 users posting #ipad in day 3. Then the ground-truth spreading size of B is 0 for day 1, 10 for day 2, and 20 for day 3.

After a user's spreading sizes over different topics in a particular period of time are computed, we can obtain the average spreading size over all topics in that time period by dividing the number of involved topics:

$$DS_u^{t_1 \sim t_2} = \frac{\sum_\tau DS_{u,\tau}^{t_1 \sim t_2}}{\#(\tau)}, \tag{5.14}$$

where DS is the spreading size, u denotes a user, $t_1 \sim t_2$ indicates a time window, and τ is a topic.

Baselines

To our best knowledge, this is the first attempt to predict the continuous-time spreading coverage of social network users. Therefore, we employ the *Autoregres-*

sive Integrated Moving Average (ARIMA) model [Mil91], which is widely used for fitting and forecasting the time series data in the area of statistics and econometrics, as one baseline. This model can first fit to time series data (in our case, a user's spreading sizes of different days in the history), then predict this user's spreading size in the future. Thus, the spreading sizes of first 12 days are used to build the ARIMA model. Then, it predicts the entire 22 days. Note that the optimal ARIMA is always selected based on Akaike information criterion (AIC) and Bayesian information criterion (BIC) for comparison.

In addition to ARIMA, one of the basic information diffusion models – *Independent Cascade* (IC) [KKT03] is used as the second baseline. In the IC model, a user u who mentions a topic at the current time step t is treated as a new activated user. An activated user has one chance to "activate" each of his/her neighbors (i.e., make them adopt this topic) with a certain probability. If a neighbor v posts the same topic after t, then he/she becomes active at time step $t+1$. Once v becomes active at time step $t+1$, u cannot activate v in the subsequent rounds.

In order to apply the IC model to calculate users' spreading sizes, the activation probability for every pair of users needs to be estimated. Specifically, for a user u, we first obtain the spreading size of each of his/her topics during the first 12 days, thus we can get average spreading size over all of his/her topics. Then, the daily average spreading size (DDS) is computed from dividing the average spreading size by 12 days. Finally, $1/DDS$ is taken as the activation probability of u and each of his/her neighbors.

Besides the abovementioned two baselines, we also compare our IDM-CTMP with two recent works from Goyal et al. [GBL10], and Saito et al. [SKOM10b] (please refer to Section 5.2 for introduction to these two works). We name these two methods as "Goyal-model" and "Saito-model" respectively.

Methods	MAE	RMSE	MASE
IDM-CTMP	3.290	4.231	0.714
ARIMA	4.369	5.470	1.294
IC	5.858	7.209	2.355
Goyal-model	4.831	6.112	1.818
Saito-model	4.412	5.861	1.773

Table 5.2: The comparison over the 10,000 top users.

Methods	MAE	RMSE	MASE
IDM-CTMP	1.686	2.055	0.702
ARIMA	2.026	2.855	0.764
IC	3.928	4.834	2.091
Goyal-model	3.130	4.118	1.987
Saito-model	2.817	4.005	1.629

Table 5.3: The comparison over the 10,000 random users.

Prediction

To compare the performance of IDM-CTMP, we choose $10,000$ top users computed by IDM-CTMP given the entire 22 days and $10,000$ random users. Three well-known metrics for measuring prediction accuracy are utilized in our experiment for evaluation: MAE (Mean Absolute Error), RMSE (Root Mean Square Error), and MASE (Mean Absolute Scaled Error).

The average values of three metrics for IDM-CTMP, ARIMA, IC, Goyal-model, and Saito-model are listed in Table 5.2 and Table 5.3. It can be seen that our proposed method IDM-CTMP performs (1) better than baseline methods ARIMA and IC, because ARIMA fits the overall trend of the time series data and does not consider the underlying network cascading causing the change of the spreading sizes. The basic IC model needs predefined time step, which is set to be 1 day. It might be too large to capture the real-time topic propagation. However, if setting it to be small, it would take long time to run. The parameter estimation of the basic IC model assumes the constant activation probability for all neighbors, which could be another reason of poor performance; (2) better than Goyal-model and Saito-model mainly because it models dynamic probabilities instead of static ones.

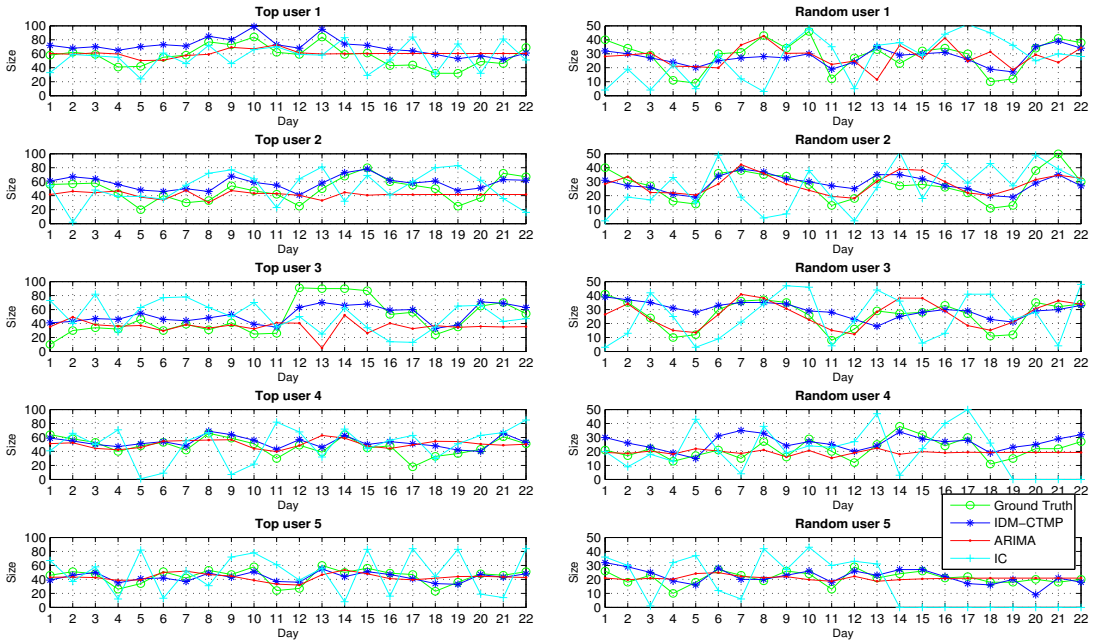


Figure 5.7: The comparison between the predicted spreading size of top ranked 5 users (left side) and randomly picked 5 users (right side) by IDM-CTMP and baseline against the ground truth.

In Figure 5.7, we plot the ground truth spreading sizes and the predicted spreading sizes of different models for both top 5 users and 5 random users. Note that the plot is mainly for illustrating how prediction results of IDM-CTMP fit the ground truth. In order to make it more readable, we skip the results of Goyal-model and Saito-model. We can observe that even though the predicted results by IDM-CTMP are not exactly same as the ground truth, most predicted curves fit very close to the true curves. In particular, most of the "peaks" and "valleys" can be well captured by our proposed method. However, ARIMA and IC does not perform well, missing many "peaks" and "valleys" and having wrong predictions.

5.7 Conclusion

In this chapter, we propose IDM-CTMP, an information diffusion model based on Continuous-Time Markov Process. IDM-CTMP is able to predict the influence dynamics of social network users, i.e., it can predict the spreading coverage of a

user within a given period of time. We also define two other dynamic influence metrics, and empirically compare different influence metrics on three dimensions of influence: i). *Monomorphism vs. Polymorphism*; ii). *High Latency vs. Low Latency*; and iii). *Information Inventor vs. Information Spreader*. Our experiment results show that the IDM-CTMP metric favors the users with high inventing ability, high spreading ability, and low topic adoption latency. In addition, IDM-CTMP achieves very promising performance as its predicted spreading size demonstrated can fit closely to the ground truth.

CHAPTER 6

CONCLUSION

6.1 Summary

This dissertation presents methods and algorithms to mine the online social network data, including organizing both users and user-generated contents, summarizing textual contents, identifying influential users. In order to organize users and contents, this dissertation proposes a novel Hierarchical Co-Clustering algorithm to simultaneously cluster both users and their contents into a tree structure. For summarizing textual contents, this dissertation considers four types of summarization tasks, and builds a submodularity-based summarization framework to perform these tasks. In addition, it proposes a novel summarization method — storyline generation, an event detection framework, and a multi-task multi-label classification method — MTML to summarize the time-sensitive contents from social media, e.g., Tweets. To identify influential users, this dissertation first introduces three important dimensions of influence, and then presents a dynamic influence model to calculate users' current influence, and predict their future influence, finally identify influential users based on their influence score.

1. Organization of users and contents

   Considering the intrinsic relationship of users and contents generated by themselves or assigned to them, as well as the hierarchy hidden in the contents, co-clustering and hierarchical clustering methods are expected to help organize the users and contents in the social network. However, results from these two separate clustering methods are difficult to interpret together, and these two methods have to be executed sequentially, or respectively using multi-thread/process machines. A novel hierarchical co-clustering algorithm, which fuses the benefits of both hierarchical clustering and co-clustering without sacrificing the interpretation ability of the results, is proposed in

this dissertation. One could execute a single algorithm once instead of running two algorithms, respectively. Furthermore, the final clustering results are more interpretable in the sense that both users, contents, and their relationships are maintained in the results.

2. Summarization of contents

   Online social network is believed to be a new media, from which audience can obtain the first-hand information about various topics. Summaries of the time-sensitive information in OSN would greatly help audience capture the big picture about their interested topics. There are many classical multi-document summarization methods. However, few of them can fulfill various requirements coming from different audiences. Furthermore, most of the existing works focus on their own domains (e.g., news, reviews) instead of the time-sensitive social media domain. In order to address these two problems, this dissertation (1) designs a summarization framework based on Submodular Functions to deal with Generic, Query-Focused, Update, and Comparative summarization. Because many other summarization problems can be mapped to Submodular Functions, this framework can naturally be extended for addressing those problems. (2) This dissertation proposes a storyline-generation method to extract summaries from the time-sensitive social media data, such as Tweets. With this method, timestamp information is well-maintained; meanwhile, audience can easily see different branches of the generated "story", and effectively capture the big picture of it. Moreover, an event detection framework and a multi-task multi-labels classification method are proposed to detect or classify new contents in a timely fashion.

3. Identifying influential users

   Identifying influential users is an important research problem in the social media domain. Traditional research is usually aiming at finding out one

single pre-defined type of "influential users", while the identified influential users might not be useful for the other purpose. Another limitation of traditional research in social media is it lacks a influence model, which can dynamically and continuously predict the influential users. This dissertation overcomes these two limitations in two-fold: (1) It introduces three important dimensions of "social influence", including Monomorphism VS. Polymorphism, High Latency VS. Low Latency, Information Inventor VS. Information Spreader, and conducts a systematically comparison over these three dimensions to reveal the true colors of "influence". This is helpful in the sense that people has guidance about how to select a method of identifying influential users akin to his/her purpose. (2) It presents a novel dynamic influence model based on Continuous-Time Markov Process to predict the influential users dynamically and continuously.

6.2 Future Work

In order to apply the Hierarchical Co-Clustering algorithm to organize the real-world social network data composed of users and contents, running time could be a big issue. The current time complexity of the Hierarchical Co-Clustering algorithm is $O(n^2 log\ n)$, which is intolerable for the large-scale social network data. To address this issue, three solutions are considered. (1) Executing the algorithm over the large-scale data offline, and then using the resulted organization tree for the online information retrieval and recommendation requests. The downside of this solution is static assumption of the input data set. (2) Implementing the algorithm in a distributed computing environment. It makes sense, but the time complexity issue remains. (3) In stead of considering every user and every piece of content, only part of them are input to the algorithm. The question lies in how to select that "part" of users and contents. The answer to this question could be random sampling some users and contents, or applying influence models to pick

up most influential users and contents. The shortcoming of this method is the sacrifice of completeness of the resulted tree.

Storyline-generation has been shown to be a reasonable way to generate and present the summary about topics in the time-sensitive social network data. However, a benchmark is missing in the evaluation part. People are eager for a canonical way or a well-accepted quantitive measurement to evaluate the quality of the resulted storyline. Following this direction would be interesting. Besides the evaluation issue, the storyline-generation is a batch algorithm. In other words, the story must be generated periodically. Assuming the life span of a hot topic is quite long, whenever an audience requests a summary of this topic, the backend engine must generate a storyline based on all historical data. As more audiences are coming continuously, the oldest data will be considered again and again, and the backend engine will take longer and longer time to finish one execution. A natural solution to this problem is re-design the storyline-generation algorithm to be an on-line version, then the story of one topic will be updated continuously.

The three dimensions of social influence can guide the others to the appropriate influence models. However, human discretion must be involved. It would be of great help if a tool, which could automatically recommend a ranking list of influence models based on various requirements about the influential users.

The proposed dynamic influence model has been shown to predict influential users (who are able to quickly diffuse topics to many other adopters) in the future. It is actually targeting on a combination of multiple types of influential users, including Low Latency users, Information Inventors, and Information Spreaders. The problem is not everyone wants "All-Around Athlete". If a tuning mechanism can be injected into the proposed dynamic influence model, people can find out four different types of influential users using one single model.

BIBLIOGRAPHY

[AA05] Lada A. Adamic and Eytan Adar. How to search a social network. *Social Networks*, 27:2005, 2005.

[AJ91] W.J. Anderson and W. James. *Continuous-time Markov chains: An applications-oriented approach*, volume 7. Springer-Verlag New York, 1991.

[All02] James Allan. Introduction to topic detection and tracking. In *Topic detection and tracking*, pages 1–16. Springer, 2002.

[ALTY08] N. Agarwal, H. Liu, L. Tang, and P.S. Yu. Identifying the influential bloggers in a community. In *WSDM*, pages 207–218, 2008.

[APL98] J. Allan, R. Papka, and V. Lavrenko. On-line new event detection and tracking. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 37–45. ACM, 1998.

[AXV+11] Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. Sentiment analysis of twitter data. In *Proceedings of the Workshop on Languages in Social Media*, pages 30–38. Association for Computational Linguistics, 2011.

[BBM04] M. Bilenko, S. Basu, and R. J. Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004.

[BDW08] S. Basu, I. Davidson, and K. L. Wagstaff. *Constrained clustering: Advances in algorithms, theory, and applications*. Chapman & Hall/CRC, 2008.

[Ber06] P. Berkhin. A survey of clustering data mining techniques. *Grouping Multidimensional Data*, pages 25–71, 2006.

[BF01] K. Barnard and D. Forsyth. Learning the semantics of words and pictures. *Computer Vision, IEEE International Conference on*, 2:408, 2001.

[BHMW11] E. Bakshy, J.M. Hofman, W.A. Mason, and D.J. Watts. Everyone's an influencer: quantifying influence on twitter. In *WSDM*, pages 65–74, 2011.

[BK07] Nilesh Bansal and Nick Koudas. Blogscope: a system for online analysis of high volume text streams. In *Proceedings of the 33rd international conference on Very large data bases*, pages 1410–1413. VLDB Endowment, 2007.

[BN08] K. Bade and A. Nürnberger. Creating a cluster hierarchy under constraints of a partially known hierarchy. In *Proceedings of the*

2008 SIAM International Conference on Data Mining, pages 13–24. Citeseer, 2008.

[BS10] Adam Bermingham and Alan F Smeaton. Classifying sentiment in microblogs: is brevity an advantage? In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1833–1836. ACM, 2010.

[CCBL12] Freddy Chong Tat Chua, William W Cohen, Justin Betteridge, and Ee-Peng Lim. Community-based classification of noun phrases in twitter. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1702–1706. ACM, 2012.

[CDGS04] H. Cho, I. S. Dhillon, Y. Guan, and S. Sra. Minimum sum-squared residue co-clustering of gene expression data. In *Proceedings of the fourth SIAM international conference on data mining*, pages 114–125. Citeseer, 2004.

[CHBG10] M. Cha, H. Haddadi, F. Benevenuto, and K.P. Gummadi. Measuring user influence in twitter: The million follower fallacy. In *ICWSM*, 2010.

[CKP93] D.R. Cutting, D.R. Karger, and J.O. Pedersen. Constant interaction-time scatter/gather browsing of very large document collections. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 126–134. ACM, 1993.

[CKPT92] D. R. Cutting, D. R. Karger, J. O. Pedersen, and J. W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 318–329. ACM, 1992.

[CL04] Madeira S. C. and Oliveira A. L. Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Transactions on Computation Biology and Bioinformatics*, 1(1):24–45, 2004.

[CL09] C.M. Chen and C.Y. Liu. Personalized e-news monitoring agent system for tracking user-interested chinese news events. *Applied Intelligence*, 30(2):121–141, 2009.

[CMN08] A. Clauset, C. Moore, and M.E.J. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453(7191):98–101, 2008.

[CNN+10] J. Chen, R. Nairn, L. Nelson, M. Bernstein, and E. Chi. Short and tweet: experiments on recommending content from information streams. In *CHI*, pages 1185–1194, 2010.

[Cun02] H. Cunningham. Gate, a general architecture for text engineering. *Computers and the Humanities*, 36(2):223–254, 2002.

[Dan07] H.T. Dang. Overview of DUC 2007. In *Document Understanding Conference*, pages 1–10, 2007.

[Der07] E.W. Dereszynski. A probabilistic model for anomaly detection in remote sensor streams. 2007.

[Dhi01] I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 269–274. ACM, 2001.

[DHP07] C. Dwyer, S.R. Hiltz, and K. Passerini. Trust and privacy concern within social networking sites: A comparison of facebook and myspace. In *Proceedings of the Thirteenth Americas Conference on Information Systems*, pages 1–12, 2007.

[DM06] H. Daumé and D. Marcu. Bayesian query-focused summarization. In *Annual Meeting-Association for Computational Linguistics*, volume 44, page 305, 2006.

[DMM03] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98. ACM, 2003.

[DO08] H.T. Dang and K. Owczarzak. Overview of the tac 2008 update summarization task. In *Proceedings of text analysis conference*, pages 1–16, 2008.

[DR01] Pedro Domingos and Matt Richardson. Mining the network value of customers. In *SIGKDD*, pages 57–66. ACM, 2001.

[DR09] I. Davidson and SS Ravi. Using instance-level constraints in agglomerative hierarchical clustering: theoretical and empirical results. *Data Mining and Knowledge Discovery*, 18(2):257–282, 2009.

[Dun73] J. C. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Cybernetics and Systems*, 3(3):32–57, 1973.

[DVA09] N. Dimililer, E. Varoğlu, and H. Altınçay. Classifier subset selection for biomedical named entity recognition. *Applied Intelligence*, 31(3):267–282, 2009.

[DYB+07] J. Davitz, J. Yu, S. Basu, D. Gutelius, and A. Harris. ilink: Search and routing in social networks. In *SIGKDD*, 2007.

[Dyn65] Evgeniĭ Borisovich Dynkin. *Markov processes*. Springer, 1965.

[Efr10] Miles Efron. Hashtag retrieval in a microblogging environment. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 787–788. ACM, 2010.

[Efr11] Miles Efron. Information search and retrieval in microblogs. *Journal of the American Society for Information Science and Technology*, 62(6):996–1008, 2011.

[EG11] Miles Efron and Gene Golovchinsky. Estimation methods for ranking recent information. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 495–504. ACM, 2011.

[EO93] T. Eckes and P. Orlik. An error variance approach to two-mode hierarchical clustering. *Journal of Classification*, 10(1):51–74, 1993.

[ER04] G. Erkan and D.R. Radev. Lexpagerank: Prestige in multi-document text summarization. In *Proceedings of EMNLP*, volume 4, 2004.

[ESBB98] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences of the United States of America*, 95(25):14863–14868, 1998.

[Fel08] Willliam Feller. *An introduction to probability theory and its applications*, volume 2. John Wiley & Sons, 2008.

[FU01] J. Foote and S. Uchihashi. The beat spectrum: A new approach to rhythm analysis. *Multimedia and Expo, IEEE International Conference on*, 0:224, 2001.

[FWE03] B. C. M. Fung, K. Wang, and M. Ester. Hierarchical document clustering using frequent itemsets. In *Proceedings of the SIAM International Conference on Data Mining*, pages 59–70, 2003.

[G+84] C. Gérard et al. Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the Rado-Edmonds theorem. *Discrete Applied Mathematics*, 7(3):251–274, 1984.

[Gar85] C.W. Gardiner. *Handbook of stochastic methods*. Springer Berlin, 1985.

[GBL10] Amit Goyal, Francesco Bonchi, and Laks V.S. Lakshmanan. Learning influence probabilities in social networks. In *WSDM*, pages 241–250, 2010.

[GD11] S. Gilpin and I. Davidson. Incorporating sat solvers into hierarchical clustering algorithms: an efficient and flexible approach. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1136–1144. ACM, 2011.

[GGLNT04] D. Gruhl, R. Guha, D. Liben-Nowell, and A. Tomkins. Information diffusion through blogspace. In *Proceedings of the 13th international conference on World Wide Web*, pages 491–501. ACM, 2004.

[GH06] Jennifer Golbeck and James Hendler. Inferring binary trust relationships in web-based social networks. *ACM Transactions on Internet Technology*, 6:497–529, 2006.

[GKRT04] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *Proceedings of the 13th international conference on World Wide Web*, pages 403–412. ACM, 2004.

[GL10] Rumi Ghosh and Kristina Lerman. Predicting influential users in online social networks. *CoRR*, abs/1005.4882, 2010.

[GLD00] G. Getz, E. Levine, and E. Domany. Coupled two-way clustering analysis of gene microarray data. *Proceedings of the National Academy of Sciences of the United States of America*, 97(22):12079–12084, 2000.

[GMCK00] J. Goldstein, V. Mittal, J. Carbonell, and M. Kantrowitz. Multi-document summarization by sentence extraction. In *NAACL-ANLP 2000 Workshop on Automatic summarization*, pages 40–48. Association for Computational Linguistics, 2000.

[GN02] M. Girvan and M.E.J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[GS96] W. Gaul and M. Schader. A new algorithm for two-mode clustering. In *Data analysis and information systems*. Springer, 1996.

[HA07] M. Hosseini and H. Abolhassani. Hierarchical co-clustering for web queries and selected urls. *Web Information Systems Engineering–WISE 2007*, pages 653–662, 2007.

[HBS10] John Hannon, Mike Bennett, and Barry Smyth. Recommending twitter users to follow using content and collaborative filtering approaches. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 199–206. ACM, 2010.

[HP96] M.A. Hearst and J.O. Pedersen. Reexamining the cluster hypothesis: scatter/gather on retrieval results. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 76–84. ACM, 1996.

[HV09] A. Haghighi and L. Vanderwende. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics on ZZZ*, pages 362–370. Association for Computational Linguistics, 2009.

[HYHN04] Q. Huang, Q. Yang, J.Z. Huang, and M.K. Ng. Mining of web-page visiting patterns with continuous-time markov models. *PAKDD*, pages 549–558, 2004.

[IPM09] D. Ienco, R. Pensa, and R. Meo. Parameter-free hierarchical co-clustering by n-ary splits. *Machine Learning and Knowledge Discovery in Databases*, pages 580–595, 2009.

[JC97] J. J. Jiang and D. W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. *Computing Research Repository*, 1997.

[JMK+00] D. Jurafsky, J.H. Martin, A. Kehler, K. Vander Linden, and N. Ward. *Speech and language processing*. Prentice Hall New York, 2000.

[JYZ+11] Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. Target-dependent twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 151–160. Association for Computational Linguistics, 2011.

[KKT03] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[KKT05] D. Kempe, J. Kleinberg, and É. Tardos. Influential nodes in a diffusion model for social networks. *Automata, Languages and Programming*, pages 99–99, 2005.

[Kle03] Jon Kleinberg. Bursty and hierarchical structure in streams. *Data Mining and Knowledge Discovery*, 7(4):373–397, 2003.

[KLPM10] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, pages 591–600. ACM, 2010.

[KMN99] S. Khuller, A. Moss, and J.S. Naor. The budgeted maximum coverage problem. *Information Processing Letters*, 70(1):39–45, 1999.

[KMS04] Ravi Kumar, Uma Mahadevan, and D Sivakumar. A graph-theoretic approach to extract storylines from search results. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 216–225. ACM, 2004.

[KRS] N. Kanhabua, S. Romano, and A. Stewart. Identifying relevant temporal expressions for real-world events.

[KT75] S Karlin and HM Taylor. A first course in stochastic processes. *Academic Press New York*, pages 474–502, 1975.

[LAD+02] Victor Lavrenko, James Allan, Edward DeGuzman, Daniel LaFlamme, Veera Pollard, and Stephen Thomas. Relevance models for topic detection and tracking. In *Proceedings of the second international conference on Human Language Technology Research*, pages 115–121. Morgan Kaufmann Publishers Inc., 2002.

[LAP+09] Theodoros Lappas, Benjamin Arai, Manolis Platakis, Dimitrios Kotsakos, and Dimitrios Gunopulos. On burstiness-aware search for document sequences. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 477–486. ACM, 2009.

[LB10] H. Lin and J. Bilmes. Multi-document Summarization via Budgeted Maximization of Submodular Functions. In *NAACL/HLT*, 2010.

[LC05] A.V. Leouski and W.B. Croft. An evaluation of techniques for clustering search results. Technical report, DTIC Document, 2005.

[LGL+08] Y. Liu, B. Gao, T.Y. Liu, Y. Zhang, Z. Ma, S. He, and H. Li. Browserank: letting web users vote for page importance. In *SIGIR*, pages 451–458, 2008.

[Lin98] D. Lin. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning*, volume 1, pages 296–304. Citeseer, 1998.

[Lin04] C.Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*, pages 25–26, 2004.

[LKG+07] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, page 429. ACM, 2007.

[LL08] Fu-ren Lin and Chia-Hao Liang. Storyline-based summarization for news topic retrospection. *Decision Support Systems*, 45(3):473–490, 2008.

[LL10] Jingxuan Li and Tao Li. Hcc: a hierarchical co-clustering algorithm. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 861–862, New York, NY, USA, 2010. ACM.

[LLL11] J. Li, L. Li, and T. Li. Mssf: a multi-document summarization framework based on submodularity. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 1247–1248. ACM, 2011.

[LLL12a] J. Li, L. Li, and T. Li. Multi-document summarization via submodularity. *Applied Intelligence*, pages 1–11, 2012.

[LLL+12b] Chen Lin, Chun Lin, Jingxuan Li, Dingding Wang, Yang Chen, and Tao Li. Generating event storylines from microblogs. In *CIKM*, pages 175–184, New York, NY, USA, 2012. ACM.

[LLO10] J. Li, T. Li, and M. Ogihara. Hierarchical Co-Clustering of Artists and Tags. In *Proceedings of the Eleventh International Society for Music Information Retrieval Conference (ISMIR 2010)*, pages 861–862. ACM, 2010.

[LMF+07] J. Leskovec, M. McGlohon, C. Faloutsos, N. Glance, and M. Hurst. Cascading behavior in large blog graphs. *arXiv preprint arXiv:0704.2803*, 2007.

[LNK07] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

[LO06] T. Li and M. Ogihara. Toward intelligent music information retrieval. *Multimedia, IEEE Transactions on*, 8(3):564–574, 2006.

[LOL03] T. Li, M. Ogihara, and Q. Li. A comparative study on content-based music genre classification. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 282–289. ACM, 2003.

[LPLS13] Jingxuan Li, Wei Peng, Tao Li, and Tong Sun. Social network user influence dynamics prediction. In *Web Technologies and Applications*, pages 310–322. Springer, 2013.

[LPN+11] Kathy Lee, Diana Palsetia, Ramanathan Narayanan, Md Mostofa Ali Patwary, Ankit Agrawal, and Alok Choudhary. Twitter trending topic classification. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pages 251–258. IEEE, 2011.

[LS00] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2000.

[LS01] B. Logan and A. Salomon. A content-based music similarity function. *Cambridge Research Labs-Tech Report*, 2001.

[LSLO12] J. Li, B. Shao, T. Li, and M. Ogihara. Hierarchical co-clustering: a new way to organize the music data. *Multimedia, IEEE Transactions on*, 14(2):471–481, 2012.

[LWZY06] B. Long, X. Wu, Z. M. Zhang, and P. S. Yu. Unsupervised learning on k-partite graphs. In *SIGKDD*, pages 317–326. ACM, 2006.

[MAH95] B. Mirkin, P. Arabie, and L.J. Hubert. Additive two-mode clustering: the error-variance approach revisited. *Journal of Classification*, 12(2):243–263, 1995.

[Man01] I. Mani. Automatic summarization. *Computational Linguistics*, 28(2), 2001.

[MC86] G. W. Milligan and M. C. Cooper. A study of the comparability of external criteria for hierarchical cluster analysis. *Multivariate Behavioral Research*, 21(4):441–458, 1986.

[Mil91] T.C. Mills. *Time series techniques for economists*. Cambridge Univ Pr, 1991.

[Min78] M. Minoux. Accelerated greedy algorithms for maximizing submodular set functions. *Optimization Techniques*, pages 234–243, 1978.

[MK10] Michael Mathioudakis and Nick Koudas. Twittermonitor: trend detection over the twitter stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 1155–1158. ACM, 2010.

[MTdRW11] Kamran Massoudi, Manos Tsagkias, Maarten de Rijke, and Wouter Weerkamp. Incorporating query expansion and quality indicators in searching microblog posts. In *Advances in Information Retrieval*, pages 362–367. Springer, 2011.

[MY04] Satoshi Morinaga and Kenji Yamanishi. Tracking dynamics of topic trends using a finite mixture model. In *SIGKDD*, pages 811–816. ACM, 2004.

[MZ05] Qiaozhu Mei and ChengXiang Zhai. Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 198–207. ACM, 2005.

[Nas08] V. Nastase. Topic-driven multi-document summarization with encyclopedic knowledge and spreading activation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 763–772. Association for Computational Linguistics, 2008.

[NBFH11] Kyosuke Nishida, Ryohei Banno, Ko Fujimura, and Takahide Hoshide. Tweet classification by data compression. In *Proceedings of the 2011 international workshop on DETecting and Exploiting Cultural diversiTy on the social web*, pages 29–34. ACM, 2011.

[New06] M.E.J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

[NHF12] Kyosuke Nishida, Takahide Hoshide, and Ko Fujimura. Improving tweet stream classification by detecting changes in word probability. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 971–980. ACM, 2012.

[NW81] GL Nemhauser and LA Wolsey. Maximizing submodular set functions: formulations and analysis of algorithms. *Studies on Graphs and Discrete Programming*, 11:279–301, 1981.

[NWF78] GL Nemhauser, LA Wolsey, and ML Fisher. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, 14(1):265–294, 1978.

[oSUC+01] National Institute of Standards, Technology (US), C. Croarkin, P. Tobias, and C. Zey. *Engineering statistics handbook*. The Institute, 2001.

[PA+98] R. Papka, J. Allan, et al. On-line new event detection using single pass clustering. *University of Massachusetts, Amherst*, 1998.

[Pha01] D. L. Pham. Spatial models for fuzzy clustering. *Computer Vision and Image Understanding*, 84(2):285 – 297, 2001.

[RD02] Matthew Richardson and Pedro Domingos. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 61–70, 2002.

[Res95] P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *In Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 448–453, 1995.

[RGAH10] Daniel M. Romero, Wojciech Galuba, Sitaram Asur, and Bernardo A. Huberman. Influence and passivity in social media. *CoRR*, abs/1008.1253, 2010.

[RJST04] D.R. Radev, H. Jing, M. Styś, and D. Tam. Centroid-based summarization of multiple documents. *Information Processing & Management*, 40:919–938, 2004.

[RMK11] D.M. Romero, B. Meeder, and J. Kleinberg. Differences in the mechanics of information diffusion across topics: idioms, political hashtags, and complex contagion on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 695–704. ACM, 2011.

[Rog03] Everett M Rogers. *Diffusion of Innovations*, volume 27. Free Press, 2003.

[SBC03] H. Saggion, K. Bontcheva, and H. Cunningham. Robust generic and query-based summarisation. *Proceedings of the European Chapter of Computational Linguistics (EACL), Research Notes and Demos*, 2003.

[SCHT07] X. Song, Y. Chi, K. Hino, and B.L. Tseng. Information flow modeling based on diffusion rate for prediction and ranking. In *WWW*, pages 191–200, 2007.

[SDRL06] A. Schlicker, F. S. Domingues, J. Rahnenfuhrer, and T. Lengauer. A new measure for functional similarity of gene products based on Gene Ontology. *BMC bioinformatics*, 7(1), 2006.

[SG03] A. Strehl and J. Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, 3:583–617, 2003.

[SJ04] J. Steinberger and K. Jezek. Using latent semantic analysis in text summarization and summary evaluation. In *Proc. ISIM04*, pages 93–100. Citeseer, 2004.

[SKC11] David A Shamma, Lyndon Kennedy, and Elizabeth F Churchill. Peaks and persistence: modeling the shape of microblog conversations. In *Proceedings of the ACM 2011 conference on Computer supported cooperative work*, pages 355–358. ACM, 2011.

[SKOM10a] Kazumi Saito, Masahiro Kimura, Kouzou Ohara, and Hiroshi Motoda. Efficient estimation of cumulative influence for multiple activation information diffusion model with continuous time delay. In *PRICAI*, pages 244–255. Springer-Verlag, 2010.

[SKOM10b] Kazumi Saito, Masahiro Kimura, Kouzou Ohara, and Hiroshi Motoda. Generative models of information diffusion with asynchronous timedelay. *Journal of Machine Learning Research - Proceedings Track*, 13:193–208, 2010.

[SLO08] B. Shao, T. Li, and M. Ogihara. Quantify music artist similarity based on style and mood. In *WIDM '08: Proceeding of the 10th ACM workshop on Web information and data management*, pages 119–124, New York, NY, USA, 2008. ACM.

[SM00] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905, 2000.

[SOM10] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *WWW*, pages 851–860. ACM, 2010.

[SR62] R. R. Sokal and F. J. Rohlf. The comparison of dendrograms by objective methods. *Taxon*, 11(2):33–40, 1962.

[SSRZG04] M. Steyvers, P. Smyth, M. Rosen-Zvi, and T. Griffiths. Probabilistic author-topic models for information discovery. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 306–315. ACM, 2004.

[STLS06] Xiaodan Song, Belle L. Tseng, Ching-Yung Lin, and Ming-Ting Sun. Personalized recommendation driven by information flow. In *SIGIR*, pages 509–516, 2006.

[TC02] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *Speech and Audio Processing, IEEE transactions on*, 10(5):293–302, 2002.

[TLT+11] Chenhao Tan, Lillian Lee, Jie Tang, Long Jiang, Ming Zhou, and Ping Li. User-level sentiment analysis incorporating social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1397–1405. ACM, 2011.

[TRM11] Jaime Teevan, Daniel Ramage, and Merredith Ringel Morris. # twittersearch: a comparison of microblog search and web search. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 35–44. ACM, 2011.

[TSK+06] P.N. Tan, M. Steinbach, V. Kumar, et al. *Introduction to data mining*. Pearson Addison Wesley Boston, 2006.

[TSWY09] J. Tang, J. Sun, C. Wang, and Z. Yang. Social influence analysis in large-scale networks. In *SIGKDD*, pages 807–816, 2009.

[TYC09] J. Tang, L. Yao, and D. Chen. Multi-topic based Query-oriented Summarization. In *Proceedings of SDM*, 2009.

[TYO11] Hiroya Takamura, Hikaru Yokono, and Manabu Okumura. Summarizing a document stream. In *Advances in Information Retrieval*, pages 177–188. Springer, 2011.

[WF94] S. Wasserman and K. Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[WLJH10] J. Weng, E.P. Lim, J. Jiang, and Q. He. Twitterrank: finding topic-sensitive influential twitterers. In *WSDM*, pages 261–270, 2010.

[WLLH08] Furu Wei, Wenjie Li, Qin Lu, and Yanxiang He. Query-sensitive mutual reinforcement chain and its application in query-oriented multi-document summarization. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 283–290. ACM, 2008.

[WLZD08] D. Wang, T. Li, S. Zhu, and C. Ding. Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 307–314. ACM, 2008.

[WTE+01] M.M. Wagner, F.C. Tsui, J.U. Espino, V.M. Dato, D.F. Sittig, R.A. Caruana, L.F. McGinnis, D.W. Deerfield, M.J. Druzdzel, and D.B. Fridsma. The emerging science of very early detection of disease outbreaks. *Journal of Public Health Management and Practice*, 7(6):51–59, 2001.

[WWL+11] Xiaolong Wang, Furu Wei, Xiaohua Liu, Ming Zhou, and Ming Zhang. Topic sentiment analysis in twitter: a graph-based hashtag sentiment classification approach. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1031–1040. ACM, 2011.

[WWS+09] F. Wang, X. Wang, B. Shao, T. Li, and M. Ogihara. Tag integrated multi-label music style classification with hypergraph. *Proc. 10th International Society for Music Information Retrieval*, pages 363–368, 2009.

[WYX07a] X. Wan, J. Yang, and J. Xiao. Manifold-ranking based topic-focused multi-document summarization. In *Proceedings of IJCAI*, pages 2903–2908, 2007.

[WYX07b] X. Wan, J. Yang, and J. Xiao. Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction. In *Annual Meeting-Association for Computational Linguistics*, volume 45, page 552, 2007.

[WZLD09] Dingding Wang, Li Zheng, Tao Li, and Yi Deng. Evolutionary document summarization for disaster management. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 680–681. ACM, 2009.

[WZLG09] D. Wang, S. Zhu, T. Li, and Y. Gong. Comparative document summarization via discriminative sentence selection. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1963–1966. ACM, 2009.

[XC99] J. Xu and W.B. Croft. Cluster-based language models for distributed retrieval. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 254–261. ACM, 1999.

[XLG03] W. Xu, X. Liu, and Y. Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, SIGIR '03, pages 267–273, New York, NY, USA, 2003. ACM.

[XM06] G. Xu and W. Y. Ma. Building implicit links from content for forum search. In *SIGIR*, pages 300–307, New York, NY, USA, 2006. ACM.

[YC10] J. Yang and S. Counts. Predicting the speed, scale, and range of information diffusion in twitter. *Proc. ICWSM*, 2010.

[YGVS07] Wen-tau Yih, Joshua Goodman, Lucy Vanderwende, and Hisami Suzuki. Multi-document summarization by maximizing informative content-words. In *IJCAI*, volume 2007, page 20th, 2007.

[YPC98] Y. Yang, T. Pierce, and J. Carbonell. A study of retrospective and on-line event detection. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 28–36. ACM, 1998.

[YWO+11] Rui Yan, Xiaojun Wan, Jahna Otterbacher, Liang Kong, Xiaoming Li, and Yan Zhang. Evolutionary timeline summarization: a balanced optimization framework via iterative substitution. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 745–754. ACM, 2011.

[ZKF05] Y. Zhao, G. Karypis, and U. Fayyad. Hierarchical clustering algorithms for document datasets. *Data Mining and Knowledge Discovery*, 10(2):141–168, 2005.

[ZL11] L. Zheng and T. Li. Semi-supervised hierarchical clustering. In *Proceedings of 2011 IEEE International Conference on Data Mining*. IEEE, 2011.

[ZLLC11] Dan Zhang, Yan Liu, Richard D Lawrence, and Vijil Chenthamarakshan. Transfer latent semantic learning: Microblog mining with less supervision. In *AAAI*, 2011.

VITA

JINGXUAN LI

January 6, 1986.	Born, Jiangsu, P.R.China
2008	B.S., Engineering Jiangsu University of Science and Technology Jiangsu, P.R.China
2008–2014	Doctoral Candidate, Computer Science Florida International University Miami, Florida

PUBLICATIONS AND PRESENTATIONS

Huang, Shu, Wei Peng, Jingxuan Li, and Dongwon Lee. "Sentiment and topic analysis on social media: a multi-task multi-label classification approach." In Proceedings of the 5th Annual ACM Web Science Conference, pp. 172-181. ACM, 2013.

Li, Jingxuan, Wei Peng, Tao Li, Tong Sun, Qianmu Li, and Jian Xu. "Social Network User Influence Sense-Making and Dynamics Prediction." Expert Systems With Applications, 2014 (Accepted).

Li, Jingxuan, Wei Peng, Tao Li, and Tong Sun. "Social Network User Influence Dynamics Prediction." In Web Technologies and Applications, pp. 310-322. Springer Berlin Heidelberg, 2013.

Li, Jingxuan, Lei Li, and Tao Li. "Multi-document summarization via submodularity." Applied Intelligence 37, no. 3 (2012): 420-430.

Li, Jingxuan, Bo Shao, Tao Li, and Mitsunori Ogihara. "Hierarchical co-clustering: a new way to organize the music data." Multimedia, IEEE Transactions on 14, no. 2 (2012): 471-481.

Li, Jingxuan, Lei Li, and Tao Li. "Mssf: a multi-document summarization framework based on submodularity." In Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval, pp. 1247-1248. ACM, 2011.

Li, Jingxuan, Tao Li, and Mitsunori Ogihara. "Hierarchical co-clustering of music artists and tags." In 11th International Society for Music Information Retrieval Conference. Utrecht, Netherlands:[sn], pp. 249-254. 2010.

Li, Jingxuan, and Tao Li. "HCC: a hierarchical co-clustering algorithm." In SIGIR, pp. 861-862. 2010.

Li, Lei, Wenting Lu, Jingxuan Li, Tao Li, Honggang Zhang, and Jun Guo. "Exploring Interaction Between Images and Texts for Web Image Categorization." In FLAIRS Conference. 2011.

Lin, Chen, Chun Lin, Jingxuan Li, Dingding Wang, Yang Chen, and Tao Li. "Generating event storylines from microblogs." In Proceedings of the 21st ACM international conference on Information and knowledge management, pp. 175-184. ACM, 2012.

Wenting Lu; Jingxuan Li; Tao Li; Weidong Guo; Honggang Zhang; Jun Guo, "Web Multimedia Object Classification Using Cross-Domain Correlation Knowledge," Multimedia, IEEE Transactions on, vol.15, no.8, pp.1920,1929, Dec. 2013.

Lu, Wenting, Lei Li, Jingxuan Li, Tao Li, Honggang Zhang, and Jun Guo. "A multimedia information fusion framework for web image categorization." Multimedia Tools and Applications (2012): 1-34.

Wu, Keshou, Lei Li, Jingxuan Li, and Tao Li. "Ontology-enriched multi-document summarization in disaster management using submodular function." Information Sciences 224 (2013): 118-129.

Zeng, Chunqiu, Yexi Jiang, Li Zheng, Jingxuan Li, Lei Li, Hongtai Li, Chao Shen et al. "FIU-Miner: a fast, integrated, and user-friendly system for data mining in distributed environment." In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 1506-1509. ACM, 2013.