Real-time Event Analysis in Online Social Networks

by

Hansu Gu

B.S., Fudan University, China, 2008

M.S., University of Colorado at Boulder, 2011

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Electrical, Computer and Energy Engineering

2013

This thesis entitled:
Real-time Event Analysis in Online Social Networks
written by Hansu Gu
has been approved for the Department of Electrical, Computer and Energy Engineering

_____

Prof. Dirk Grunwald

_____

Prof. Fabio Somenzi

Date _____

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Gu, Hansu (Ph.D., Computer Engineering)

Real-time Event Analysis in Online Social Networks

Thesis directed by Prof. Dirk Grunwald

Online social networks(OSNs) enable real-time event discussion. Due to the word-of-mouth effect, popular events are disseminated exponentially in a short period of time. With highly active public engagement, new events are being self-reported and discussed live. Compared to traditional news event detection and tracking, this huge volume of data, unstructured content, and variety of information in OSNs pose both opportunities and challenges for event analysis in new environments. This thesis makes key contributions in the following three aspects.

Event context identification helps to answer the question of who is interested in the events. It enables applications like user participation prediction, relevant event recommendation and friendship recommendation. We incorporate anchor information into the traditional probability matrix factorization framework to identify the group of users who are interested in given event. Our evaluation based on one-month of 461 events and 1.1 million users shows that our approach outperforms at least 20% over existing approaches.

Location inference addresses the problem of lacking location information in event analysis. It helps to understand where the event is being discussed. We use both textual and structural information to predict locations respectively, and finally use a learn-to-rank algorithm to effectively fuse the results. Evaluation a three-month of 0.82 million users, 16.4 million messages, and 11.5 million friendships shows the performance boost of 25% reduction in average error, and 66% reduction in median error over existing work.

Event modeling provides a solution for understanding the structure of the event. We first build a hierarchical and incremental model for each event, and then identify the causal relationships within the event structure. Our evaluation on 3.5 million messages over a 5-month period and demonstrate the high effectiveness and efficiency of our approach.

Dedication

To my family.

Acknowledgements

I would like to acknowledge all the help I received to finish my PhD thesis.

First of all, I would like to express my deepest gratitude to my advisor Prof. Dirk Grunwald for his guidance and help throughout my PhD study. His vision, knowledge and insights inspire me strongly to find valuable research ideas. His passion, patience and trust greatly encourage me to dig deeply into the problems. I always feel fortunate to work in the research environment built by Prof. Grunwald.

I would also like to thank my committee members especially Prof. Qin Lv, Prof. Fabio Somenzi and Dr. Yaoping Ruan for all their constructive suggestions and comments on my thesis. Their advice always greatly help me to think through my research problems. I owe particular gratitude to Prof. Lv, who is very generous to share her broad vision and keen insights on my thesis topic, as well as help guide my research work in detail.

It is also my great honor to receive help from my collaborators, colleagues and friends. I would like to thank Mike Gartrell, Liang Zhang, Haojie Hang, Aaron Schram, Chenyu Zheng, Lei Tian, Ning Gao, Yifei Jiang, Yu-li Liang, for all the research discussions and collaborative work. Special appreciation is due to my BFIS friends Richard Kiefer and Patricia Butler, who help me smoothly adapt to the culture and life in Boulder. My PhD life is healthier and more diverse thanks to the support from Longmont Table Tennis Club, especially my practice partners Jacques Middlecoff and David Vollmar.

Last and most importantly, I want to thank my parents, whose endless support and love keep encouraging me to overcome all the difficulties. To them I dedicate this thesis.

This thesis was sponsored in part by the National Science Foundation under grant #IIS-0910586.

Contents

Chapter

1 Introduction **1**

1.1 Event Context Identification 4

1.2 Location Identification 4

1.3 Event Modeling 5

2 Related Work **6**

2.1 Event Context Identification 6

2.2 Location Inference 8

2.3 Event Modeling 9

3 Event Context Identification **11**

3.1 Introduction 11

3.2 Problem Formulation and System Overview 13

3.2.1 Definitions and Problem Formulation 13

3.2.2 System Overview 14

3.2.3 Preliminary: PMF 14

3.3 The AnchorMF Model 15

3.3.1 Anchor Selection 15

3.3.2 Incorporating Anchor Information into the PMF Framework 19

3.3.3 Model Inference 21

3.3.4 Implementation Details and Computational Complexity Analysis . . . 23
3.4 Experimental Evaluation . . . 25
3.4.1 Experimental Setup . . . 25
3.4.2 Prediction Performance . . . 27
3.4.3 Event Context Case Study . . . 29
3.4.4 Retrieval of Relevant Events based on Event Context . . . 33
3.4.5 Friendship and Event Context . . . 34
3.5 Conclusions . . . 36

4 Location Inference **38**
4.1 Introduction . . . 38
4.2 Problem Formulation and System Overview . . . 40
4.2.1 Geographic Regions . . . 42
4.3 Text-based Location Ranking . . . 43
4.3.1 Relevance Ranking Based on Textual Features . . . 43
4.3.2 Geo-sensitive Textual Features . . . 44
4.4 Structure-based Location Ranking . . . 45
4.4.1 MLE-based Ranking Framework . . . 46
4.4.2 Friendship Modeling and Parameter Estimation . . . 46
4.5 Fusion (Re-ranking) . . . 49
4.5.1 Properties of Our Re-ranking Problem . . . 49
4.5.2 Re-ranking Algorithm Design . . . 50
4.6 Experimental Evaluation . . . 53
4.6.1 Experimental Setup . . . 53
4.6.2 Overall Quality . . . 56
4.6.3 Quality of Text-based Location Ranking . . . 59
4.6.4 Quality of Structure-based Location Ranking . . . 60

4.6.5 Quality of Re-ranking . . . 62
4.7 Discussions . . . 62
4.8 Conclusions . . . 64

5 Event Modeling **65**
5.1 Introduction . . . 65
5.2 Problem Formulation and System Overview . . . 67
5.3 Information Block Identification . . . 68
5.3.1 N-Gram Based Block Identification . . . 69
5.3.2 Conservation-Based Block Identification . . . 71
5.4 Incremental Hierarchical Theme Structure Construction . . . 71
5.4.1 Static Theme Structure Construction . . . 72
5.4.2 Incremental Structure Construction . . . 74
5.5 Causal Relationship Detection . . . 75
5.6 Experimental Evaluation . . . 78
5.6.1 Dataset Description . . . 79
5.6.2 Quality of Information Blocks and Theme Structures . . . 79
5.6.3 Efficiency of Incremental Event Modeling . . . 80
5.6.4 Quality of Detected Causal Relationships . . . 81
5.6.5 Case Study . . . 81
5.7 Conclusions . . . 82

6 Applications and Usage Scenarios **88**
6.1 Event Context Identification . . . 88
6.1.1 Event Correlation . . . 88
6.1.2 User Recommendation . . . 89
6.1.3 Friendship Recommendation . . . 89
6.2 Location Inference . . . 90

6.2.1 Event Discussion Location . . . 90
6.2.2 Event Reaction . . . 90
6.3 Event Modeling . . . 90
6.3.1 Event Story Analysis . . . 91
6.3.2 Event Tracking and Prediction . . . 91
6.4 Usage Scenarios . . . 91
6.4.1 Company Brand Construction . . . 91
6.4.2 Disaster Response . . . 92

7 Conclusion **94**
7.1 Summary . . . 94
7.2 Future Directions . . . 95
7.2.1 Richer Feature Set . . . 95
7.2.2 Larger Scale . . . 96

Bibliography **97**

Appendix

Tables

Table

3.1 Number of user pairs and their average number of shared anchors and shared events. 17

3.2 Correlation between #anchors and #events per user. 18

3.3 Event categories. 26

3.4 Data statistics. 26

3.5 Prediction performance comparison. 28

3.6 Event case study: #nola 30

3.7 Event case study: #hbgsmc 30

3.8 Event case study: #2013ces 31

3.9 Event case study: #oscarnoms 31

3.10 Event case study: #lakers 32

3.11 Event case study: liverpool 32

3.12 Event retrieval ranking results. 34

4.1 Combination of Different Textual Feature 44

4.2 Data Set Properties 56

4.3 Overall Quality Comparison 56

4.4 Results of Top 10 Most Populous Counties 58

4.5 Re-rank Results 62

5.1 List of Events Used in Evaluation 77

Figures

Figure

1.1 Overview of Event-centric Research Problems . . . 3

3.1 AnchorMF system overview. . . . 14

3.2 Probabilistic matrix factorization (PMF). . . . 15

3.3 Anchor and user distribution. . . . 16

3.4 AnchorMF graphical model. . . . 19

3.5 Speedup with the increasing number of cores . . . 25

3.6 Event size distribution. . . . 27

3.7 CDF of user-friend similarity based on event context. . . . 35

3.8 Friend similarity distribution per user. . . . 36

4.1 GeoFind: Fusing textual and social structure information in online social networks to geolocate users. . . . 41

4.2 Probability of Friendship vs. Distance in log-log scale. Dotted line is the fitted power-law distribution. . . . 47

4.3 Distribution of 200 geographic regions (clusters) obtained from data set 2. Bigger circles represent more users. . . . 54

4.4 Distribution of 18-month of all geo-tagged tweets from Twitter firehose. Brighter color (darker in grey color mode) represents more users. . . . 55

4.5 Conditional regression tree showing the estimation error under different situations, e.g., Node 14 with high estimation accuracy. 57

4.6 Parameter Tuning on Number of Regions 59

4.7 Quality comparison of GeoFind structure-only and FindMe. 61

4.8 Efficiency comparison of GeoFind structure-only and FindMe. 61

4.9 GeoFind: Text-only vs. structure-only vs. re-ranking. 63

5.1 ETRee: Effective and efficient event modeling for real-time online social media networks — System overview. 83

5.2 Two cases demonstrating when ETree outperforms TSCAN. 84

5.3 Event 7: Comparison of computation time of ETree, ETree-NI and TSCAN. 85

5.4 Event 12: Comparison of computation time of ETree, ETree-NI and TSCAN. 85

5.5 Event 19: Comparison of computation time of ETree, ETree-NI and TSCAN. 85

5.6 Causal relationship pairs detected by ETree and TSCAN 86

5.7 Partial hierarchical theme structures constructed by ETree for the event “Haitian earthquake”. 87

Chapter 1

Introduction

With the rapid development of Web 2.0 technologies, Online Social Networks(OSNs) have been increasingly popular. From traditional online forums, blogs, to more recent Facebook, Twitter, various kinds of OSNs are making social interactions much more convenient. Online forums and blogs make it easy for social discussion and opinion exchange. To name a few, Facebook, Twitter and Google+ help people to better follow their friends' update and enable faster communication. Foursquare and Yelp enable location-based services for better local interactions. Among all the characteristics OSNs have, we notice that the real-time characteristic is very unique. It is one of the reasons that attract so many people to choose OSNs as their favorite social interaction platforms. Furthermore, it makes possible to turn OSNs from social interaction platforms to live content providers. People's immediate updates, discussions and opinions can be very informative and considered as personal social media. Thanks to the real-time user participation, a huge amount of social media content is being generated by individual users in an instant fashion. For instance, users of Twitter [69, 80], a popular microblogging social media site, send over 400 million tweets per day [68][1] , meaning 4629 tweets per second on average. On the other hand, with the growing ubiquity of GPS-enabled mobile devices, location information is becoming prevalent in today's OSNs. As another dimension of social networking, users can geotag their posts, and announce their current locations to friends and the public at large. For instance, on Twitter, users are allowed to attach their location information to any message they send. Foursquare encourages users to check in with their location and they have collected more than 4.5 billion checkins in total [67][2] . The unique opportunity of anytime, anywhere user participation

[1] Statistics in March, 2013.
[2] Statistics in Oct, 2013

makes it possible for event-centric analysis of OSNs.

Given this large amount of real-time, location enabled, user generated content, OSN platforms can be viewed in a latent event-centric perspective. For instance on Twitter, despite the fact that users use the platform for daily chats and conversations with friends, research shows that users even intended to share information/URLs and to report events on Twitter [37]. Twitter shows significant non-social characteristics and 85% of the trending topics on Twitter are related to headline news in nature [43]. As events happen and evolve over time, users stay informed by seeking and sharing information through their social contacts (e.g., "following" and "follower" networks in Twitter). As a result, OSNs have become the online gathering place for public engagement when real-time events happen and offer valuable self-reported information about events. This has been demonstrated in various application domains, such as disease surveillance [19], hazardous situations [65], new product release, presidential campaign, so on and so forth. By sharing and receiving information among trusted and/or close social contacts, information related to specific events can be generated and disseminated in a highly effective and efficient fashion. Twitter also enables the capability of social filtering, in the sense that users are receiving recommendations filtered by their friends(e.g. retweets) and they can recommend the same content again if they feel appropriate. This is a particular important characteristic for efficient event dissemination. As shown in [43], once retweeted, a tweet gets retweeted almost instantly on Twitter for the next couple of hops, a fascinating way of speeding up the diffusion of events.

State-of-the-art research mainly focuses on event detection in OSNs [77] [35] [14] [56] [47] [73]. Event detection techniques provide an automatic way of generating the important topics/keywords given real-time message stream. For instance, one can quickly follow ongoing *Haitian Earthquake* events by extracting keywords like *haitian, haiti, earthquake, magnitude*, etc. Essentially, event detection techniques help to identify real-time events, but provide no further structure to the events. As a result, there are some unsolved research problems in order to get event-driven perspectives of OSNs.

At the same time, existing infrastructures do not fully support the event centric perspective and have the following missing components. Firstly, the context of an event is usually undefined. Given a detected popular event, it is essentially hard to define the group of users who participate in the event. As a result, it is even more difficult to characterize the users if they share similar interests or are from similar locations.

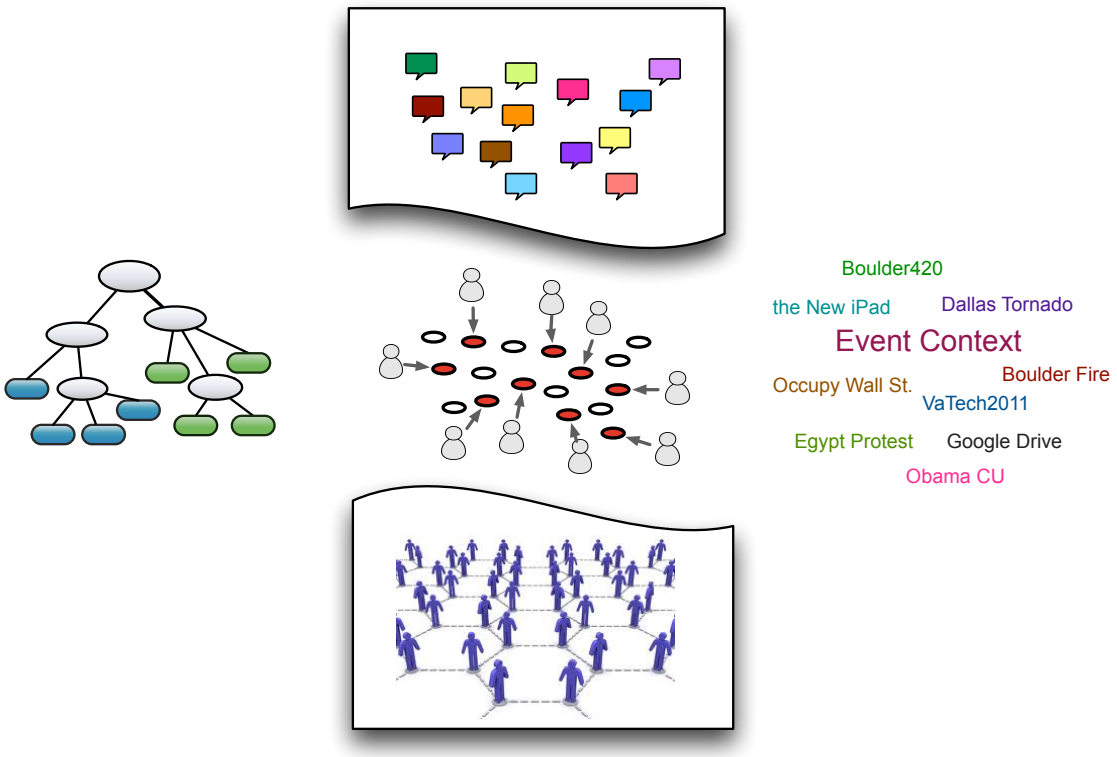

Figure 1.1: Overview of Event-centric Research Problems

Secondly, despite the usefulness, there is a relatively small percentage of location data compared to the non-geotagged data volume. The analysis of our own dataset shows that there are only 0.5% to 1% geotagged content currently on Twitter. This results in unknown information of where the events are being discussed. Thirdly, user generated content is presented in an unstructured format and it is very difficult for users to read these flat-viewed, self-reported messages. In most cases, during an event, users will never have the chance to read them all. Overall, none of the three aspects are being supported in current OSNs.

Based on above observations, we identify and solve the following research problems in support of event-driven knowledge discovery in real-time OSNs. An overview of the research problems is shown in figure 1.1.

1.1 Event Context Identification

The goal of event context identification problem in OSNs is to define the group of users who participate in a certain event. Recent research [65] [64] has shown that event discussions in OSNs are diverse and innovative and encourage public engagement in events. Although much research has been conducted on OSNs to track and detect events, there has been limited research on detecting or understanding the **event context**. Event context identification helps to predict users' participation of in events, identify relations among events, and recommend friends who share similar event context. We propose a matrix factorization based technique that aims to identify event context by leveraging a prevalent feature in OSNs, the **anchor** information. Our work makes three key contributions: (1) a formal definition of the event context identification problem; (2) anchor selection and incorporation into the matrix factorization process for effective event context identification; and (3) demonstration of applying event context for user-event participation prediction, relevant events retrieval, and friendship recommendation. Evaluation based on 1.1 million Twitter users over a one-month data collection period shows that our event content identification algorithm achieves a 20.0% improvement in terms of user-event participation prediction.

1.2 Location Identification

We explore location identification problem in order to find where the events happen. Location information is becoming important in today's online social networks. However, the amount of available data is not much. We are trying to find a solution, with which even when no explicit location is disclosed by a user, it is still possible to geolocate the user through his/her social context, e.g., status updates and social relationships in OSNs. To demonstrate this, we accurately identifies users' geographic regions through effective fusion (re-ranking) of (1) text-based ranking using geo-sensitive textual features and (2) structure-based ranking using maximum likelihood estimation (MLE) of geotagged friends. Evaluation results using 0.8 million geotagged Twitter users over a 3-month period demonstrate that our approach outperforms state-of-the-art techniques, with significant reduction of estimation error (25% of average error, 66% of median error). As a side effect, the potential of improving location accuracy through the fusion of multiple data

types calls for a re-examination of existing privacy protection policies and mechanisms.

1.3 Event Modeling

Effective event modeling helps understand the content of the event. Realtime online social networks such as Twitter provide great opportunities for public engagement and event information dissemination. Event-related discussions occur in real time and at the worldwide scale. However, these discussions are in the form of short, unstructured messages and dynamically woven into daily chats and status updates. Compared with traditional news articles, the rich and diverse user-generated content raises unique new challenges for tracking and analyzing events. Effective and efficient event modeling is thus essential for real-time information-intensive OSNs. We propose such a solution for social media network sites. Targeting the unique challenges of this problem, our solution consists of three key components: (1) an n-gram based content analysis technique for identifying core information blocks from a large number of short messages; (2) an incremental and hierarchical modeling technique for identifying and constructing event theme structures at different levels of granularity; and (3) an enhanced temporal analysis technique for identifying inherent causalities between information blocks. Detailed evaluation using 3.5 million tweets over a 5-month period demonstrates that our approach can efficiently generate high-quality event structures and identify inherent causal relationships with high accuracy.

Chapter 2

Related Work

Event-driven knowledge discovery aims to fill the gap between social interaction perspective to event-centric perspective. It draws upon research in several related fields, including traditional news event summarization, evolution and detection, joint analysis of location and web content, and Twitter related research.

2.1 Event Context Identification

Event detection and tracking. There has been much event-related research in the literature. The field of *event detection and tracking* can be traced back to [77] [3] [76]. Kleinberg defined and extracted bursts of activity from emails using an infinite-state automaton [40]. Bursty events can also be detected from news texts by identifying bursty features with a binomial distribution model and threshold-based heuristics [26]. Ihler et al. focused on time-series data such as logs and proposed Markov-Poisson models to detect anomalous events [35]. A general probabilistic model was proposed to extract correlated bursty topic patterns in [71]. Chen et al. used user tag information to identify events that involve browsing and searching photos on Flickr [14]. Lappas et al. explored how bursty terms help enhance the search process [45]. These works show the importance and effectiveness of event analysis using Web data.

Event analysis on Twitter. More research has been conducted on Twitter recently. Different crisis events have been analyzed to identify generative and innovative properties of discussion on Twitter [44] [65] [64]. Sakaki et al. developed an earthquake alarm system by extracting real-time earthquake events on Twitter [60]. Petrović et al. presented a locality sensitive hashing approach to efficiently detect events that have not been seen before based on tweets [56]. Weng et al. proposed wavelet-based signal clustering

on Twitter text stream data to detect events [73]. Lin et al. leveraged interests of users and social relations to track the evolution of popular events [47]. Event popularity can be predicted by considering a variety of social features [31]. Such event detection techniques support algorithmic discovery of events on OSNs, and help to build the foundation of event-related research. However, they do not solve the event context identification problem directly.

Matrix factorization techniques. This work also builds upon existing matrix factorization techniques. Salakhutdinov et al. proposed a probabilistic matrix factorization model, which factorizes the explicit user-item matrix to a user latent trait matrix and an item latent trait matrix [62]. A full Bayesian version of the model was also proposed to provide generalized parameter tuning and avoid overfitting [61]. For implicit datasets, Hu et al. adopted more features from the original user-item matrix and proposed an improved gradient descent method to solve the problem more efficiently [33]. More recent research considers friendship information as a useful feature to incorporate into the current framework. Trust based approaches consider friendship as trust to influence users' latent factors. In [49], friendship information was modeled as a linear combination of the basic model. Another approach was also proposed to model friends as a separate latent matrix and used friendship as observations [50]. SoicalMF was proposed to incorporate friendship into the same latent space as users and a user's latent factor is represented as the average of all friends' latent factors [36]. Gartrell et al. proposed to consider only close friends when combining friends' latent factors and used a Markov random field to aggregate latent factors [27]. Influence based models consider users' interests to be influenced by their friends. Huang et al. considered receiver interests, item qualities, and interpersonal influences for final recommendation [34]. Jiang et al. incorporated interpersonal influences into the existing PMF model and showed significant performance improvement [38]. The assumption of influence-based models is that items must be coming from their friends, which is not always the case. Our work separates the anchor latent factor space from the user latent factor space with a feature selection process, which shows better performance than existing solutions.

2.2 Location Inference

In this work, we study the effectiveness of geolocating users through the fusion of textual and social structure information in OSNs, as well as its implications on location privacy. As such, our work builds upon research in several related fields, including location sharing and its privacy concerns, as well as geographically aware web mining and learn-to-rank techniques.

Joint Analysis of Location and Web Content. The joint analysis of location and web content has been an area of active research. Previous works have investigated the spatio-temporal theme patterns in weblogs [51], associations of geographic information with search engine queries [5], and relation between location and content in large photo collections [20]. Yin et al. incorporated GPS information into topic modeling to provide more accurate geographical topic discovery [79]. The relationship between location and textual features has also been explored. Volz et al. introduced an ontology-based approach to disambiguate geographical names in texts [59]. Buscaldi et al. suggested both WordNet and Naive Bayes based approaches for the automatic identification of geographical articles in encyclopedic resources [10]. More recent studies have focused on OSNs. Researchers have shown that user profile information can be inferred using social graph and friendship information [8, 53]. Backstrom et al. proposed FindMe and measured the relationship between location and friendship, and predicted the locations of individuals in Facebook [6]. Multiple techniques have been proposed to predict Twitter users' locations, including GeoLex, a cascaded topic modeling approach based on users' tweets [23], LocalTweet, a maximum-likelihood estimation (MLE)-based technique using local words in users' tweets [15], and SimpGeo, a grid-based approach that leverages Kullback-Leibler (KL) divergence [75]. In comparison, our solution focuses on location estimation through the fusion of geo-sensitive textual features and the social structures of friendships, and achieves much better accuracy.

Location sharing and its privacy concerns. Location sharing and its privacy concerns have attracted increasing attention recently. Various location sharing applications have been proposed, including friendship estimation using location trail data [21], using location sharing to improve social conversation within groups [7], and leveraging both social and geographical influences to recommend points of interest [78]. Significant privacy concerns are raised in location sharing environments. A personal daily life trail can be

inferred by analyzing location data gathered from mobile devices [1]. Kostakos et al. proposed an attack model by analyzing social network structures [42]. Anthony et al. discussed the privacy policy variation with different places and social contexts [4]. The willingness of location sharing highly depends on who was requesting [18] as well as the social context and activities [39]. While most research works assume that users understand and have full control of their location sharing, our work demonstrates that users' location information can be inferred through the company they keep, even when no explicit location information is disclosed by the users themselves. Our results may help to design better privacy control policies in the future, or to better advise users on what information should or should not be disclosed.

Learn-to-rank techniques Learn-to-rank or machine-learned ranking techniques have also been studied in recent years. The goal is to automatically construct a ranking model via (semi-)supervised learning. A systematic study of pointwise, pairwise, and listwise approaches are presented in [48]. In particular, RankBoost [24] and RankNet [9] are two widely-used techniques, but perform worse than our re-ranking algorithm (Section 4.5) when geolocating users in OSNs.

2.3 Event Modeling

This work aims to identify inherent event theme structures and causal relationships in real-time information-intensive offline social media networks. It draws upon research in several related fields, including text summarization, time-based event evolution, as well as recent research that is specific to the Twitter social media network site.

Text summarization. Much research has been conducted in the area of text summarization, focusing mostly on news data and email data. Summaries of news articles included temporal single-sentence summaries [2], centroid-based summaries of multiple documents [57] and reference relationships among distinctive phrases [46]. Fung et al. used traditional bisecting k-means clustering algorithm to model news hierarchy [25]. The obtained event hierarchy may not be meaningful since news articles are always partitioned into a fixed number of clusters (e.g., 2). Targeting email data, Carenini et al. investigated the problem of discovering important hidden emails using fragment quotation graph and generating email summaries using clue words [11]. Our solution differs from these works in that it handles short messages (tweets), gen-

erates both summaries and hierarchical theme structures without the need to specify the number of themes beforehand, and adjusts the event models incrementally as new content is continuously generated in a real-time fashion.

Time-based event evolution. This line of work focuses on the temporal changes/relationships of events. Kleinberg identified (emerging/changing) themes in document streams (e.g., emails, research papers) based on their temporal burstiness and hierarchical structure [41]. To understand how an event emerges, changes, and disappears, Subasic et al. separated each event into several stages with equal time period and represented each stage by building a network of salient terms based on their co-occurrence frequency and time relevance [66]. The problem of identifying event causal relationships has also been investigated by researchers [55, 13]. These techniques consider both content similarity and temporal proximity in order to identify possible causal relationships in events. Our work follows this rationale, but considers the more precise temporal distribution information rather than the beginning and ending time of events. As a result, our solution achieves higher precision and recall in causal relationship identification.

Other twitter specific research. Twitter has attracted much attention in the research community during the recent years. Starbird et al. analyzed the rapid generation of Twitter communications in the Red River flood event and identified generative, synthetic, derivative and innovative properties [65]. Sakaki et al. utilized tweets as social sensors to successfully detect events like earthquake or typhoon [60]. By analyzing the top trending topics, Kwak et al. found the fast information diffusion property [43]. User intentions of using Twitter's microblogging and community services have also been studied [37, 80]. A recommendation system has been built based on both content and collaborative filtering techniques [32]. Users' tweet history is used for determining their locations, enabling better personalized services [15]. These works are complementary to our work, as they did not consider the problem of event modeling at Twitter, but nevertheless provided useful insights into the various properties of Twitter.

Chapter 3

Event Context Identification

3.1 Introduction

With the rapid growth of online social networks (OSNs), more and more real-world events are being discussed on Web 2.0 platforms such as Facebook, Twitter, Tumblr, etc. Researchers have been using these platforms as social sensors to detect events, analyze event-related discussions, and predict event popularity. Despite much research on the aforementioned topics, there has been limited research that aims to detect or understand the **context** of events. The context for an event is essentially represented by the group of users who show inherent interests or willingness to participate in the event, such as people supporting their home football team, residents affected by a local fire or flooding, or people interested in Oscar nominations. The aggregated attributes of the group typically demonstrate commonalities in location, interests, age, gender, etc.

Event context identification is an important research problem and has many real-world applications. Successful event context identification will help to better predict the users who are going to participate in an event, thus creating value for enterprises and organizations for better marketing and event management. Event context also helps to identify relations among events if they share the same or similar context. Interesting patterns may be discovered even if events are not semantically related but otherwise share similar context. For example, as we will show in the experiments, event *Obama 2013 inauguration* is related to event *The International Consumer Electronics Show (CES) 2013* according to identified context. Another application is friendship recommendation based on the event context for past event participation. As we will later show in the experiments described in Section 5.6, friendships are correlated with event context

similarity among users.

Event context identification is a challenging problem for several reasons. First, it is difficult to define event context properly. Context is a subjective concept and the same group of users may be interpreted according to different common features. It is typically easier for a computer algorithm to discover a contextual pattern, rather than explain the cause for this pattern. Second, although other techniques may be applied to solve the event context identification problem, their performance is not good [62] [36]. Given historical event data, we can extract event contexts by characterizing events based on user participation, and at the same time characterizing users by their event participation. This process is very similar to the idea of *matrix factorization* [62]. Previous research mainly considered the original user-item rating matrix (i.e., the user-event matrix in our setting) and friendship information if available. However, as we show later in our experiments, friendship information does not show significant performance improvement. Finally, users interested in certain types of events tend to follow certain **anchor** accounts in OSNs. However, it is not clear how these anchor accounts can be selected (among massive following/follower relations), nor is it clear how to incorporate such anchor information into the overall event context identification process.

To address these challenges, we have developed *AnchorMF*, a unified solution for identifying event context by utilizing both user-event participation information and anchor information in OSNs. Given observations of user-event and user-follower matrices, a probabilistic model is built to consider users, events, and anchors as latent factors. An anchor selection algorithm is proposed to automatically identify informative anchors for the model. Finally, a Gibbs sampler and a maximum a posteriori (MAP) estimator are proposed to estimate the parameters of the model. AnchorMF is implemented and evaluated using a real-world Twitter data set which we have collected over one month and contains 1.1 million Twitter users. Evaluation results show that AnchorMF outperforms state-of-the-art techniques by 20.0% in terms of prediction accuracy. AnchorMF can identify relevant events using an information retrieval process. We also show that event contexts can be used for friendship recommendation.

To the best of our knowledge, this is the first work that aims to address the event context identification problem. This work makes the following contributions: (1) a formal definition of the event context identification problem; (2) anchor selection and incorporation into the matrix factorization process for effective

event context detection; and (3) application of event context to user-event participation prediction, relevant events retrieval, and friendship recommendation.

3.2 Problem Formulation and System Overview

3.2.1 Definitions and Problem Formulation

Each **event** $e = \{m_1, m_2, \ldots\}$ is represented by a set of messages obtained by searching for specific keywords $W = \{w_1, w_2, \ldots\}$ in an OSN (e.g., Twitter) and corresponds to a real-world event. Each message $m_i =< u_i, t_i >$, meaning that the message was posted by user u_i at time t_i. u_i is considered a "participant" of event e in the cyber world.

The **context** of a given event is defined as a group of users who participate in the event because of some inherent reasons, i.e., common attributes of the participants or latent event/user factors. For instance, both location and interest are important attributes to represent event context: the context of a local basketball game could be the group of local people who like their basketball team. Therefore, the context of event e_j can be jointly characterized by the event latent factor E_j and the set of user latent factors U_{e_j} of all the participants of e_j.

Anchors are popular users or public pages in OSNs, and their followers tend to participate in certain types of events, e.g., the Twitter account of a local news venue or a user posting actively on a specific topic. Usually, anchors are not directly identified by OSNs, and any user who has followers can be an anchor candidate. Selecting anchors for effective event context identification is the key. Let $\mathbb{U}_a$ be the set of followers of anchor candidate a, we select a as an anchor based on the following two factors:

(1) $|\mathbb{U}_a| \geq threshold$, i.e., the anchor must have at least $threshold$ followers. $threshold$ is set to 269 based on our modeling analysis shown in Section 3.3.

(2) The probability of a being an anchor depends on a's concentration of events $\mathbb{E}$, i.e., $\mathbb{U}_a$ participate in similar events. This probability is used as a weight in the model to reflect the impact of this anchor candidate.

The problem of **event context identification** is then defined as follows. Given M events $\mathbb{E} = \{e_1, e_2, \ldots, e_M\}$ participated by N users $\mathbb{U} = \{u_1, u_2, \ldots, u_N\}$, the output of event context identification is $C = \{c_1, c_2, \ldots, c_M\}$, where each c_i is the context of e_i. The event contexts capture the event latent factors and user latent factors, which in turn can identify the subset of users who are likely to participate in each event. The success of event context identification can be evaluated by comparing the user-event participation predicted by the event contexts and the actual user participation in events. Detailed evaluation results are presented in Section 5.6.

3.2.2 System Overview

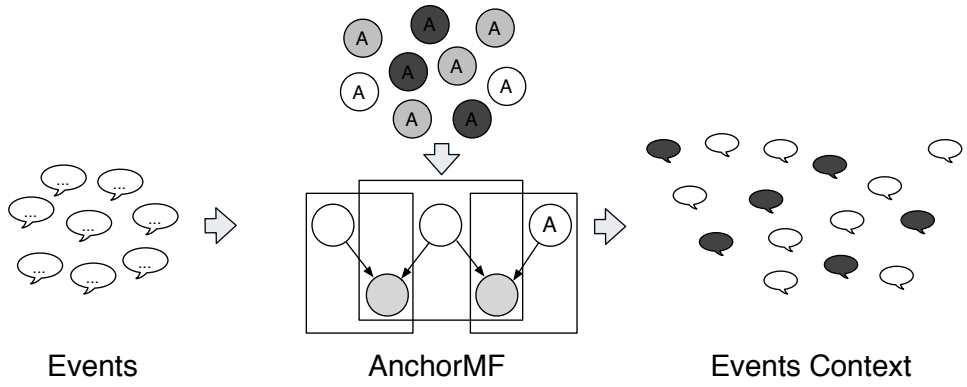

Figure 3.1: AnchorMF system overview.

Figure 3.1 illustrates the high-level process of **AnchorMF** for event context identification. Given a set of events, we first select anchors from the candidate users (Section 3.3.1), then incorporate the selected anchors into an extended probabilistic matrix factorization (PMF) model (Section 3.3.2), and finally through model inference (Section 3.3.3) we obtain the event contexts represented by event and user latent factors.

3.2.3 Preliminary: PMF

Given a set of N users $\mathbb{U} = \{u_1, \ldots, u_N\}$, a set of M events $\mathbb{E} = \{e_1, \ldots, e_M\}$, and the binary matrix $R = [R_{ui}]_{N \times M}$ representing users' participation in events, the probabilistic matrix factorization (PMF) model factorizes R into two latent matrices $U \in \mathbb{R}^{K \times N}$ and $E \in \mathbb{R}^{K \times M}$, representing k-dimensional latent

trait vectors for users and events. The graphical model is shown in Figure 3.2.

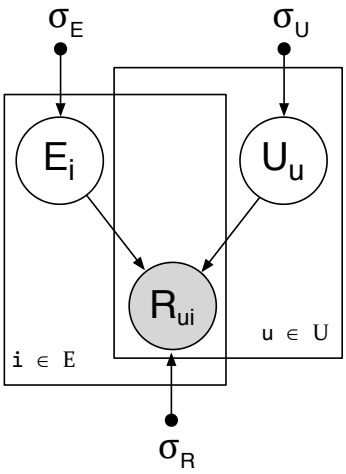

Figure 3.2: Probabilistic matrix factorization (PMF).

PMF defines the following distributions:

$$
\begin{aligned}
p(R|U,E,\sigma_R^2) &= \prod_{u=1}^{N}\prod_{i=1}^{M}\mathcal{N}(R_{u,i}|U_u^T E_i,\sigma_R^2) \\
p(U|\sigma_U^2) &= \prod_{u=1}^{N}\mathcal{N}(U_u|0,\sigma_U^2\mathbf{I}) \\
p(E|\sigma_E^2) &= \prod_{i=1}^{M}\mathcal{N}(E_i|0,\sigma_E^2\mathbf{I})
\end{aligned}
\tag{3.1}
$$

3.3 The AnchorMF Model

3.3.1 Anchor Selection

As discussed in Section 5.2, anchors are any user accounts which have at least a certain number of followers and whose followers show a good concentration on similar events. Using a real-world Twitter dataset we have collected (Section 3.4.1), we start with anchor candidates with at least 1 follower, and the set of candidate anchors shrinks as the selection process progresses. For simplicity, we refer to the anchor candidates in each round as anchors.

3.3.1.1 Anchor and User Distribution

We first need to understand the relation between anchors and their followers. The problem can be decomposed into the distribution of followers given anchors and the distribution of anchors given followers.

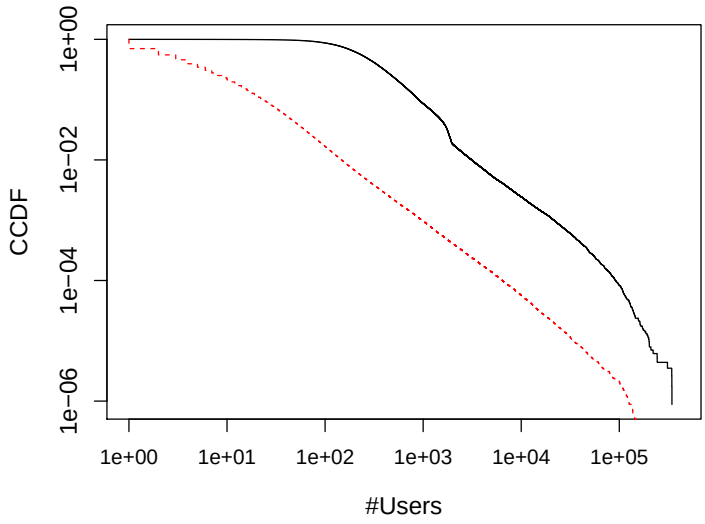

Figure 3.3: Anchor and user distribution.

The dotted curve in Figure 3.3 shows the complementary cumulative distribution function (CCDF) of the number of followers given anchors. The x-axis is the number of followers and the y-axis is the percentage of anchors. This heavy-tailed distribution shows that many anchors are followed by few users and very few anchors are followed by many users. The solid curve in Figure 3.3 shows the CCDF of the number of anchors that users follow. Here, the x-axis is the number of anchors and the y-axis is the percentage of users. This solid curve is also a heavy-tailed distribution and shows that many users follow few anchors and very few users follow many anchors. We notice that the solid curve has a flat beginning; this indicates that users tend to follow approximately 100 anchors at minimum. We also notice that there is an anomaly near 2,000 followers for the solid curve. This is likely due to the fact that Twitter's policy [70] allows each user to follow at most 2,000 anchors unless he/she is very active on Twitter. As we can see from the figure, less than 5% of the users in our dataset follow more than 2,000 anchors. The gap between the dotted and solid

#pairs(M)	16.32	7.86	4.64	3.04	3.69	2.81	4.25
#anchors	1	2	3	4	5.42	7.84	19.68
#events	2.47	2.68	2.82	2.92	3.02	3.13	3.35

Table 3.1: Number of user pairs and their average number of shared anchors and shared events.

curves is caused by the fact that when counting the number followers of anchors, we only consider the users in our event dataset, and not all followers of the anchors at Twitter.

We use the goodness-of-fit based method proposed in [16] to fit the two CCDFs shown in Figure 3.3. The power law model gives us two parameters α and x_{min}. α is the scaling parameter, which indicates how skewed the distribution is (the slope of the CCDF). As described in [16], a typical value of α is between 2 and 3. Our estimated α is 2.24 for the anchor distribution and 2.26 for the user distribution. These results match what we see in Figure 3.3 and the model shown in [43], which indicate that our dataset is representative. The second parameter, x_{min}, indicates the minimum x-axis value that fits the power law. The x_{min} of the anchor distribution is 269, and we use this number as the minimum frequency of an anchor candidate. Therefore, all the anchor candidates must have at least 269 followers. With this parameter setting, the number of anchor candidates is reduced to less than 1% of the original anchor candidate set size, which significantly reduces the amount of computation in our modeling process.

3.3.1.2 Relation Between Anchors and Events

Before considering the event concentration of anchors, we first study the relation between anchors and events, specifically, if users who follow the same anchors tend to participate in the similar events. We randomly sampled 10,000 users from our dataset, and consider for each pair of users the number of shared anchors and number of shared events. We separate all the user pairs into different buckets based on quantile and ensure that all user pairs with the same number of shared anchors fall into the same bucket. Table 3.1 shows the aggregate results for each bucket, including the number of user pairs, average number of shared anchors, and average number of shared events. As shown in the table, when users share more anchors, the number of shared events also increases. Therefore, identifying the appropriate anchors can serve as good indicators for event participation and event context identification.

#users	Pearson Correlation	Spearman Correlation
10,000	0.11	0.12

Table 3.2: Correlation between #anchors and #events per user.

We further analyze for each user if the number of anchors he/she follows is correlated with the number of events he/she participates in. We use both Pearson's correlation to check linear correlation and Spearman's correlation to check non-linear correlation. The formulas are shown in Eq. 3.2 and Eq. 3.3.

$$r_{A,E} = \frac{E[(A - \mu_A)(E - \mu_E)]}{\sigma_A \sigma_E} \tag{3.2}$$

$$\rho_{A,E} = \frac{E[(a - \mu_a)(e - \mu_e)]}{\sigma_a \sigma_e} \tag{3.3}$$

As shown in Table 3.2, there is very little correlation between a user's number of anchors and number of events, showing that one is not a substitute for the other. These results indicate that anchor information and event information are complementary signals, and adding anchor information on top of user-event information can potentially boost the performance of event context identification.

3.3.1.3 Event Concentration and Anchor Weight

Based on the anchor-user distribution analysis, we prune anchor candidates with fewer than 269 followers. Next, we need to select candidates which show a good concentration of similar events. We solve this problem by first looking at the users who follow an anchor and the events that those users participate in. We compute an anchor-event matrix by multiplying the anchor-user and user-event matrices:

$$M_{ae} = M_{au} \times M_{ue} \tag{3.4}$$

Each element $N_{ki} = M_{ae}[k, i]$ is the number of anchor k's followers who participate in event i. We denote E' as the set of events participated by anchor k's followers, and each event i is duplicated N_{ki} times in the set, i.e., $E' = \{e_{1,1}, \ldots, e_{1,N_{k1}}, \ldots, e_{M,N_{kM}}\}$. We then need to consider whether the events in E' are

similar to each other. The event concentration for each anchor k is defined as:

$$w_k = \sum_{i=1}^{M} \binom{N_{ki}}{2} \cdot 1 + \sum_{i=1, j \geqslant i+1}^{M} N_{ki} \cdot N_{kj} \cdot S(i,j) \tag{3.5}$$

The formula above aims to compute the average pair-wise event similarity for events in E', which is used to represent the anchor's concentration over events. If a pair contains two of the same events, the similarity is 1, otherwise, the similarity is defined by $S(i,j)$:

$$S(i,j) = \int_{-\infty}^{i^T j} \mathcal{N}(i^T j; 0, 1) d(i^T j) \tag{3.6}$$

The similarity function is the cumulative normal distribution of the inner product space given the i and j event pairs [27]. The intuition is that the similarity should be defined in the inner product space and lie between 0 and 1. Based on the similarity function and each anchor's event concentration, we can then select anchors and proceed with incorporating the anchor information in the AnchorMF model.

3.3.2 Incorporating Anchor Information into the PMF Framework

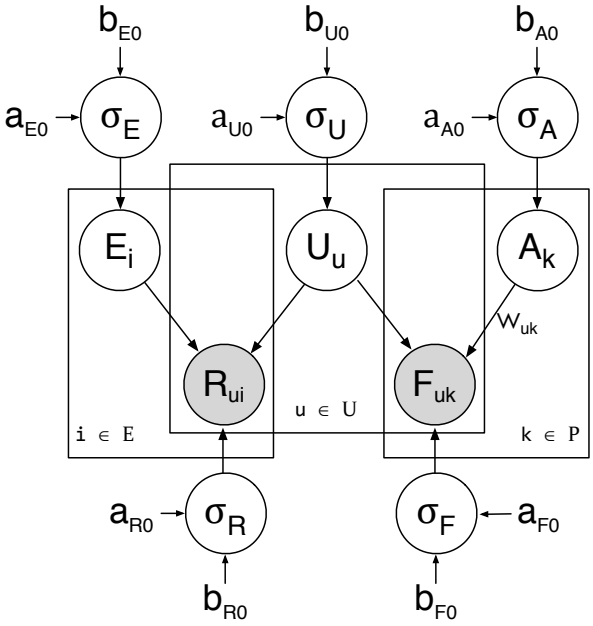

Figure 3.4: AnchorMF graphical model.

We incorporate anchor information into the PMF framework by factorizing the user-anchor matrix and also considering the importance of the anchors. As illustrated in Figure 3.4, the AnchorMF model considers a new observation F, which indicates what anchors each user follows. Correspondingly, we add a latent factor A to represent the anchors' latent influence on users. According to Equation 3.5, each anchor k has a weight, and weight is the same for every user u, denoted as W_{uk}. Consistent with the PMF model, we also add priors and hyper-priors into the model. U and E are latent variables for users and events, respectively, and R is a binary observation matrix where each element indicates whether or not a user participates in an event. We derive Equation 3.7 directly from Figure 3.4.

$$
\begin{aligned}
&p(U, E, A|R, F, \sigma_R^2, \sigma_U^2, \sigma_E^2, \sigma_A^2, \sigma_F^2) \\
&\quad \propto p(R|U, E) \times p(F|U, A) \times p(U) \times p(E) \times p(A) \\
&\quad = \prod_{u=1}^{N} \prod_{i=1}^{M} \mathcal{N}(R_{u,i}|U_u^T E_i, \sigma_R^2) \\
&\quad \times \prod_{u=1}^{N} \prod_{k=1}^{P} \left[\mathcal{N}(F_{u,k}|U_u^T A_k, \sigma_A^2)\right]^{W_{uk}} \\
&\quad \times \prod_{u=1}^{N} \mathcal{N}(U_i|0, \sigma_U^2 \mathbf{I}) \times \prod_{i=1}^{M} \mathcal{N}(E_i|0, \sigma_E^2 \mathbf{I}) \\
&\quad \times \prod_{k=1}^{P} \mathcal{N}(A_i|0, \sigma_A^2 \mathbf{I})
\end{aligned}
\tag{3.7}
$$

To facilitate model inference, we also derive the log of the posterior probability as follows:

$$
\begin{aligned}
&\ln p(U, E, A|R, F, \sigma_R^2, \sigma_U^2, \sigma_E^2, \sigma_A^2, \sigma_F^2) \\
&\quad = -\frac{1}{2\sigma_R^2} \sum_{u=1}^{N} \sum_{i=1}^{M} (R_{u,i} - U_u^T E_i)^2 \\
&\quad - \frac{1}{2\sigma_R^2} \sum_{u=1}^{N} \sum_{k=1}^{P} w_{u,k} (F_{u,k} - U_u^T A_k)^2 \\
&\quad - \frac{1}{2\sigma_U^2} \sum_{u=1}^{N} U_u^T U_u - \frac{1}{2\sigma_E^2} \sum_{i=1}^{M} E_i^T E_i - \frac{1}{2\sigma_A^2} \sum_{k=1}^{P} A_k^T A_k + C
\end{aligned}
\tag{3.8}
$$

We denote $-\frac{1}{2\sigma_R^2}$ as λ_R, $-\frac{1}{2\sigma_U^2}$ as λ_U, $-\frac{1}{2\sigma_E^2}$ as λ_E, $-\frac{1}{2\sigma_A^2}$ as λ_A, and $-\frac{1}{2\sigma_F^2}$ as λ_F. We model

$\{\lambda_R, \lambda_U, \lambda_E, \lambda_A, \lambda_F\}$ as conjugate Gamma distributions with flexible hyperpriors similar to:

$$p(\lambda_U) = \mathcal{G}(\lambda_U; a_{u0}, b_{u0}) = \frac{1}{\Gamma} b_{u0}^{a_{u0}} \lambda_U^{a_{u0}-1} e^{-b_{u0}\lambda_U} \tag{3.9}$$

3.3.3 Model Inference

Given the Bayesian framework defined in the previous section, inference for this model can be performed through Gibbs sampling [28]. Gibbs sampling generates a number of samples from an aperiodic and irreducible Markov chain, and involves sampling from the conditional distribution for each latent variable to approximate the joint distribution given that sampling from the joint distribution of the model is difficult. In the above model, we denote the random variables by $\theta = \{U, E, A, \lambda_U, \lambda_E, \lambda_A, \lambda_R, \lambda_F\}$, and we derive the following conditional distributions for all the random variables based on Equation 3.8.

λ_U is sampled from a Gamma distribution:

$$\begin{aligned} \lambda_U|\theta_{\setminus\lambda_U} &\sim \mathcal{G}(\lambda_U; a_U, b_U) \\ a_U &= a_{U0} + \frac{|\mathcal{U}|\,K}{2} \\ b_U &= b_{U0} + \frac{1}{2}\sum_{i\in\mathcal{U}} \|\mathbf{U}_i\|^2 \end{aligned} \tag{3.10}$$

λ_E and and λ_A are sampled from similar conditional distributions.

λ_R is also sampled from a Gamma distribution:

$$\begin{aligned} \lambda_R|\theta_{\setminus\lambda_R} &\sim \mathcal{G}(\lambda_R; a_R, b_R) \\ a_R &= a_{R0} + \frac{|\mathcal{R}|}{2} \\ b_R &= b_{R0} + \frac{1}{2}\sum_{u,i\in\mathcal{R}} (R_{ui} - U_u^T E_i)^2 \end{aligned} \tag{3.11}$$

We set $a_{E0} = a_{U0} = a_{A0} = \sqrt{K}$, $b_{E0} = b_{U0} = b_{A0} = 1$, and $a_{R0} = b_{R0} = a_{F0} = b_{F0} = 1.5$. The parameter settings make sure the gamma distributions are flat and allow the model to be flexible when learning.

U_u is conditionally sampled from a multivariate Gaussian distribution:

$$\begin{aligned} U_u|R,F,\theta_{\setminus U_u} &\sim \mathcal{N}(U_u;\mu_u,\Sigma_u) \\ \mu_u &= \Sigma_u(\lambda_R \sum_{i=1}^{M} R_{ui}E_i + \lambda_F \sum_{k=1}^{P} W_{u,k}F_{u,k}A_k) \\ \Sigma_u &= (\lambda_R \sum_{i=1}^{M} E_i \cdot E_i^T + \lambda_F \sum_{k=1}^{P} W_{u,k}A_k \cdot A_k^T + \lambda_U \mathbf{I})^{-1} \end{aligned} \tag{3.12}$$

E_i is conditionally sampled from a multivariate Gaussian distribution:

$$\begin{aligned} E_i|R,F,\theta_{\setminus E_i} &\sim \mathcal{N}(E_i;\mu_i,\Sigma_i) \\ \mu_i &= \Sigma_i(\lambda_R \sum_{u=1}^{N} R_{ui}U_i) \\ \Sigma_i &= (\lambda_R \sum_{u=1}^{N} U_u \cdot U_u^T + \lambda_U \mathbf{I})^{-1} \end{aligned} \tag{3.13}$$

A_k is also conditionally sampled from a multivariate Gaussian distribution:

$$\begin{aligned} A_k|R,F,\theta_{\setminus A_k} &\sim \mathcal{N}(A_k;\mu_k,\Sigma_k) \\ \mu_k &= \Sigma_k(\lambda_F \sum_{u=1}^{N} W_{u,k}F_{u,k}U_u) \\ \Sigma_k &= (\lambda_F \sum_{u=1}^{N} W_{u,k}U_u \cdot U_u^T + \lambda_U \mathbf{I})^{-1} \end{aligned} \tag{3.14}$$

The Gibbs sampling approach described above computes an approximation of the posterior distribution, which allows us to infer users' participation in events, but it does not find the maximum point of the posterior. Therefore, it is difficult to compute a point estimate of the latent matrices U and E which result in the maximum function value from the Gibbs sampling results. However, U and E describe the event context we need, and thus we need good point estimate for these variables to calculate the similarities between users and events. To this end, we also propose a maximum a posteriori (MAP) estimation for the model which estimates the maximum point (mode) of the posterior distribution, and therefore generates point estimates for U and E. The MAP estimator empirically converges faster than the Gibbs sampling approach.

MAP estimation works by maximizing the conditional distributions of U and E iteratively, where:

$$\begin{aligned} U_u &= \mu_u \\ E_i &= \mu_i \\ A_k &= \mu_k \end{aligned} \tag{3.15}$$

Since all the conditional distributions are Gaussian distributions, the Gaussian's mean will define the curvature and how far to step towards the maximum point for each iteration. This approach is similar to the inference algorithm described in [54].

The complete inference algorithm is shown in Algorithm 1 on page 24

3.3.4 Implementation Details and Computational Complexity Analysis

To save computational resources, we first consider how to reduce the complexity for sampling $U_u \sim \mathcal{N}(U_u; \mu_u, \Sigma_u)$. When calculating Σ_u, we need to compute $\lambda_U \mathbf{I}$ and $\lambda_F \mathbf{I}$. Note that these two terms are exactly the same for all the U_u samples, so they only need to be calculated once for each iteration. Also $\lambda_R \sum_i E_i e_i^T$ and $\lambda_F \sum_k A_k A_k^T$ are also independent of the samples, and these entries can also can be computed only once for each iteration. When calculating μ_u, $\lambda_R \sum_i R_{ui} E_i^T$, and $\lambda_F \sum_u F_{uk} A_k^T$, 90% of the terms have default ratings, which are independent of the samples, and need to be computed only once in each iteration. In summary, for each sample U_u only around 10% of full work for computing μ_u needs to be conducted. This is also true for each E_i and A_k sample.

Although the sampling of U_u, E_i and A_k must be conducted sequentially, the sampling of different Us, Es and As can be conducted in parallel. This saves significant computation time in practice. We implemented a parallelized MAP estimator using the thread pool mechanism in the Java standard library. Empirically, the MAP estimator converges within 100 iterations and finishes within 0.5 hours on 8 cores(2.4GHz for each core) based on our own dataset described in Section 5.6. From figure 3.5, as the number of cores increases, the speedup based on the running time of single core also increases in linear with a slope of 0.81. The efficiency is less than 1.0 because of synchronization overhead. Compared to the traditional SocialMF approach which takes 3.2 hours and is not parallelizable, AnchorMF is much more efficient and scalable.

Algorithm 1 AnchorMF

for t = 1 **to** num_of_samples **do**
  if Gibbs sampling and $t < burn\_in\_samples$ **then**
    $\{\lambda_U, \lambda_E, \lambda_A, \lambda_R, \lambda_F\} = preset\_value$
  else
    Sample $\{\lambda_U, \lambda_E, \lambda_A, \lambda_R, \lambda_F\}$ (Eq. 3.10 and Eq. 3.11)
  end if
  for users u = 1 **to** N **do**
    if Gibbs sampling **then**
      Sample U_u in parallel (Eq. 3.12)
    else if MAP **then**
      Use mean to compute U_u in parallel (Eq. 3.15)
    end if
  end for
  for events i = 1 **to** M **do**
    if Gibbs sampling **then**
      Sample E_i in parallel (Eq. 3.13)
    else if MAP **then**
      Use mean to compute E_i in parallel (Eq. 3.15)
    end if
  end for
  for anchors k = 1 **to** P **do**
    if Gibbs sampling **then**
      Sample A_k in parallel (Eq. 3.14)
    else if MAP **then**
      Use mean to compute A_k in parallel (Eq. 3.15)
    end if
  end for
end for

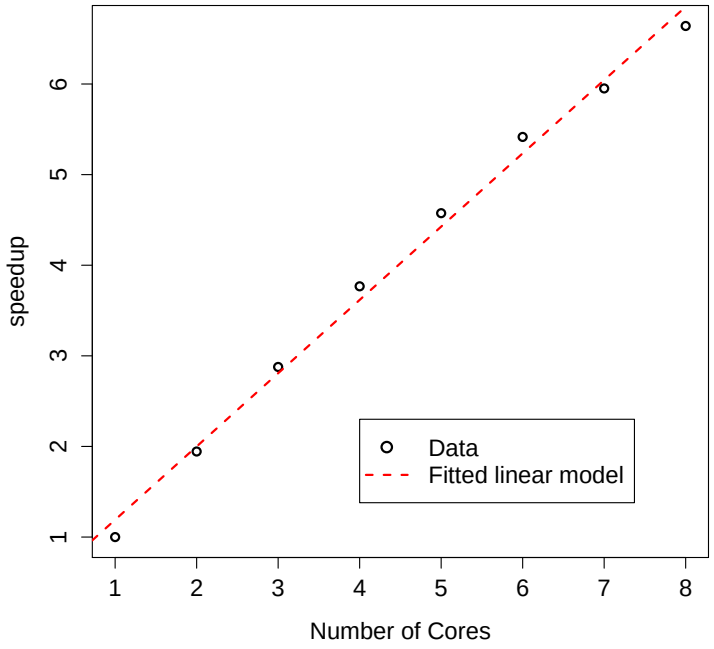

Figure 3.5: Speedup with the increasing number of cores

3.4 Experimental Evaluation

In this section, we evaluate AnchorMF, our proposed event context identification solution, using real-world events that we have collected. Our evaluation aims to answer the following questions:

- Does AnchorMF provide good predictive performance for user participation in events?
- Is the identified event context interpretable?
- Is event context effective for retrieving relevant events?
- Is event context useful for friendship recommendation?

3.4.1 Experimental Setup

We collect data using the Twitter API. We monitor daily Twitter trending topics and get a list of ranked popular keywords by considering both how long they stay on the trending topics and their rank. Then

Category	#Events
Sports	248
Entertain	134
Tech	17
Social	25
Politics	37

Table 3.3: Event categories.

Users	1.1M
Events	461
Anchors	0.59M
User-event pair	20.79M
User-achor pair	175.99M
friendship pair	92.72M

Table 3.4: Data statistics.

a human review process is used to review the top 200 keywords and identify the ones that match real-world ongoing events. The selected keywords are then filtered on real-time Twitter streams to continue collection of messages which contain the keywords. At the same time, we search for historical tweets which contain the keywords for up to 7 days. Since the selected keywords are mostly filtered on the day they became popular, we believe a 7-day look-back window is enough to collect complete events based on keywords. The data collection process introduces some noise into the dataset, but we carefully choose representative keywords to ensure events are not too general. For example, the *2013 Obama inauguration* event was collected based on the keywords *Obama inauguration*, rather than *Obama*, which tends to have a much broader scope. After we collect all the desired events, we only consider users who have participated in at least 5 of these events. This procedure helps us to remove much of the noise in the dataset. We believe most of our data consists of complete and coherent events. After obtaining the users who participated in each event, we also collect friends of the users, users' profiles, and lists (a group of followed users with a group name). In total, over a one-month period of time from Jan 4th to Feb 3rd, 2013, we collected 461 events consisting of 20.79M tweets and 1.1M users. All the data are stored in MongoDB and the total volume of the data is 554 GB. Statistics for this dataset are shown in Table 3.4. From Table 3.3 we can see that events are divided mainly into five categories. Sports and Entertainment events dominate the event type distribution. The CDF of the number of users per event is shown in Figure 3.6. We see that our event dataset consists of both large and small events and event size follows a heavy-tailed distribution.

All the experiments have been conducted on a 2.4 GHz 16-core machine with 48GB of memory. This machine runs Ubuntu 12.04.2 and JVM 1.6.0_27. All of the implementation and experiments are written in Java.

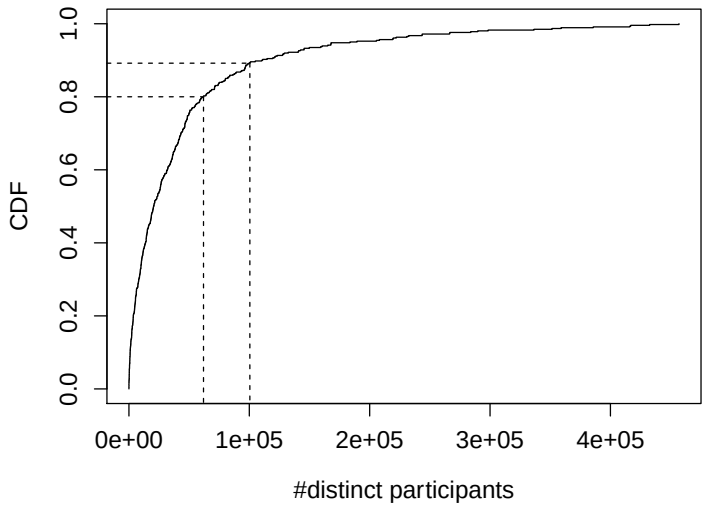

Figure 3.6: Event size distribution.

3.4.2 Prediction Performance

In this experiment we want to examine the effectiveness of our event context identification in terms of predicting user participation in events. We run 10-fold cross validation and in each fold randomly select 10% of all the events in our dataset as test events and the remaining 90% as training events. For each test event, we sort all the users according to the time they participated in the event, and use the top 10%, 20%, 30%, 40%, or 50% as training users for the test event. Our goal is to evaluate the predictions of the other 90%, 80%, 70%, 60% and 50% of users for test events.

In Table 4.3, each row represents a method of ranking test users for a test event. These methods are:

(1) *Random* ranking predicts testing users in a randomized order. This method is the baseline for prediction.

(2) *Popularity* based ranking predicts test users who are popular in the training data as ranked higher for test events. This method does not consider event context and is the baseline for contextual prediction.

Training Users	10%	20%	30%	40%	50%
Random	0.5	0.5	0.5	0.5	0.5
Popularity	0.349	0.349	0.349	0.349	0.349
Baseline PMF	0.313	0.276	0.258	0.247	0.240
SocialMF	0.313	0.277	0.258	0.246	0.240
AnchorMF-Gibbs	0.213	0.204	**0.197**	0.196	0.193
AnchorMF-MAP	**0.212**	**0.202**	**0.197**	**0.193**	**0.192**

Table 3.5: Prediction performance comparison.

(3) *Baseline* ranking makes prediction based on event context identified by the PMF matrix factorization technique [62] using only user and event information. This method serves as the baseline for prediction based on event context.

(4) *SocialMF* identifies event context based on the matrix factorization technique considering user friendship information. This method shows better performance than the baseline PMF approach [36].

(5) *AnchorMF* is the model we propose to identify event context based on the matrix factorization technique that leverages anchor information. We compare the performance of AnchorMF using either the Gibbs sampler or the MAP estimator.

We use average rank percentile as our main evaluation metric, which is a recall-based metric from [33], since the implicit dataset we use does not include complete data for precision based measurements. Users who actually participated in the events should be ranked higher in the prediction results. The average rank percentile is computed as follows:

$$\overline{rank} = \frac{\sum_e (\sum_u rank_{ue} / |u|)}{|e|}, \tag{3.16}$$

where $rank_{ue}$ is the average rank percentile for each user u in the event e. 0 represents the highest rank, while 1.0 represents the lowest rank so lower percentile means better performance.

As shown in Table 4.3, AnchorMF outperforms SocialMF by 20.0% when using 50% of the users as training users for the test events. As the percentage of training users decreases, we see a larger performance boost for AnchorMF compared to SocialMF, up to 32.2%. We also notice that for both SocialMF and baseline PMF, in the case where we only use 10% of the training data, the performance is almost as bad as

the non-contextual popularity-based method.[1] However, AnchorMF with 10% training data performs better than the best cases for both SocialMF and baseline PMF. This shows the effectiveness of the identified anchor information, and indicates that it is particularly helpful to identify event context and predict user participation in the early stage of an event. When we compare the SocialMF and the baseline PMF approaches, we do not see much performance difference for our dataset. One possible explanation, as we will see in Section 3.4.5, is that on Twitter users tend to have friends who are very dissimilar in terms of the latent trait space. Therefore the use of aggregated friends' interests, as performed in SocialMF, may not be beneficial. Additionally, since we removed users who have participated in fewer than 5 events from our dataset, and since SoicalMF has been proved to be most effective for cold start users in recommender systems, SocialMF is not effective in our scenario since there are no cold start users.

3.4.3 Event Context Case Study

As shown above, our proposed event context identification algorithm is effective and outperforms other existing related approaches. We would now like to see how to interpret the identified context. The experiment described in this subsection examines three different scenarios, where each scenario has a different event context. The results show that the identified event context is interpretable and meaningful.

We select 6 events from all predicted results we get from the experiment described in Section 3.4.2. Each event consists of the predicted users for that event; the information for each user includes Twitter profile and list data. We aggregate the user information for each event and use this data to populate a table, as shown in Tables 3.6 through 3.11. Each column represents a source or dimension of user information that we will examine, including location, the self-provided user profile description, and tags from users' list information. We extract all keywords from this aggregated user information and list the top five keywords ranked by probability of occurrence ($P = frequency\_count/total\_frequency$). We study the context of events by looking at these keywords and manually verify whether they have coherent semantic meaning.

[1] We only consider the popularity of users in the training events, so the performance of popularity based ranking is independent of the percentage of training users in testing events.

Location	P	Description	P	Tags	P
new orleans	0.42	sports	0.06	sports	0.08
LA	0.12	music	0.04	travel	0.05
Louisiana	0.05	world	0.03	politics	0.03
city	0.03	god	0.03	music	0.02
usa	0.02	football	0.03	entertain	0.01

Table 3.6: Event case study: #nola

3.4.3.1 Location Specific Event Context

First, we look at two cases where users discuss events on Twitter based on location. Table 3.6 shows results from an event about a local famous cafe that moved to a new location, which happened on Jan 17, 2013. As we can see from the results, the location dimension has a concentration of probability on the keywords *new orleans*, which matches the actual location of this event. The rest of the keywords, such as *LA* and *Louisiana* in the location dimension also have coherent meaning. Although *city* and *usa* are general location terms which do refer to a specific location, they have much lower probability compared with higher ranked keywords. If we look at both the description and tag dimensions, the keywords all have fairly low probability without much concentration, and they also lack coherent semantic meaning.

Table 3.7 shows results from an event about a local social club meetup in Harrisburg, Pennsylvania that happened on Jan 21, 2013. The results are very similar to what we see from the *#nola* event. The location dimension has a concentration and coherent meaning, while the tag dimension does not. We do see that the description dimension has the *social* keyword with higher probability. The reason for this is the type of the event is essentially a social event and people participating in the event are self-identified with the keyword "social".

Location	P	Description	P	Tags	P
pa	0.31	social	0.12	twibes	0.07
lancaster	0.12	manager	0.05	pa	0.06
harrisburg	0.12	business	0.05	social	0.04
pennsylvania	0.06	marketing	0.05	local	0.04
county	0.06	foodie	0.05	leader	0.04

Table 3.7: Event case study: #hbgsmc

Location	P	Description	P	Tags	P
new york	0.08	media	0.05	news	0.37
ca	0.08	news	0.04	media	0.13
usa	0.06	tech	0.04	tech	0.09
canada	0.04	writer	0.03	marketing	0.02
tx	0.03	marketing	0.02	business	0.02

Table 3.8: Event case study: #2013ces

3.4.3.2 Interest Specific Event Context

We now look at two examples that are based on users' interests. We focus on the description dimension and the tag dimension to see if the keywords extracted give us meaningful information.

Table 3.8 shows results from an event about the International Consumer Electronics Show from Jan 8 to Jan 11, 2013. As we can see from the results, the tag dimension has a concentration on news, media, and tech, which match the event's semantic meaning. Also as expected, the location dimension shows a broad coverage of different locations and does not have a concentration as compared to location-specific events. However, we do not see significant concentration in the description dimension, although the top keywords have coherent semantic meaning. We will discuss this result further in Section 3.4.3.3.

Table 3.9 shows results from an event about the Oscar nominations which happened on Jan 10, 2013. The results show the same pattern as what we find in the International Consumer Electronics Show event.

3.4.3.3 Location and Interest Specific Event Context

Next, we look at two events that are both location and interest specific. Good examples of these types of events are local sports events. We will focus on all of the three dimensions to see if there are any

Location	P	Description	P	Tags	P
new york	0.12	film	0.11	news	0.48
los angeles	0.06	tv	0.09	entertain	0.10
london	0.06	writer	0.09	tv	0.02
torronto	0.04	news	0.09	fashion	0.02
canada	0.03	movies	0.07	media	0.02

Table 3.9: Event case study: #oscarnoms

Location	P	Description	P	Tags	P
ca	0.15	sports	0.10	sports	0.32
los angeles	0.10	life	0.08	nba	0.08
california	0.05	love	0.08	basketball	0.03
tx	0.04	fan	0.06	fans	0.02
san antonio	0.03	music	0.03	lakers	0.02

Table 3.10: Event case study: #lakers

interesting patterns.

Table 3.10 shows the results of NBA basketball game event that involved the Los Angeles Lakers vs. the San Antonio Spurs, which happened on Jan 9, 2013. The location dimension shows an interesting concentration on both Los Angeles and San Antonio, which are the expected locations. The description dimension shows a somewhat noisy results, but the sport keyword is apparent. The tag dimension shows good concentration and gives us confidence that this is indeed a local sports event.

Table 3.11 shows the results of an event involving two English soccer teams in the Premier League, Manchester United vs. Liverpool, which happened on Jan 13 , 2013. Similar to the results of the previous example, the location and tag dimensions show the expected results, while the description is relatively noisy.

From all these event case studies, we find that it is relatively easy for humans to understand the event context by looking at the location and tag dimensions. This results from the fact that the location field is specifically designed for users to provide their location on Twitter, and most people tend to follow this rule. Tags are provided by users' followers and they serve identification purposes, and so tend to include location and interest information. The results also indicate the usefulness of the textual data in these dimensions and may lead us to incorporate this information into our model in the future. We also notice that the description

Location	P	Description	P	Tags	P
london	0.16	fan	0.10	football	0.14
uk	0.08	football	0.07	sports	0.13
England	0.07	sports	0.05	sport	0.07
manchester	0.05	united	0.04	soccer	0.05
liverpool	0.02	arsenal	0.04	friends	0.03

Table 3.11: Event case study: liverpool

dimension is noisy for all three types of events, because self descriptions are very informal and users do not usually include their location and interest information in their self-provided profile description.

3.4.4 Retrieval of Relevant Events based on Event Context

By looking at the common attributes of predicted users for some events, we can understand the meaning of event context. The next important issue to investigate is how we can use the identified context to better understand events. In this experiment, our goal is to demonstrate the feasibility of building an event-based search engine by leveraging event context information. The challenge here is that we do not consider the text of events and the relevance is only based on event context.

We have built a proof-of-concept system to evaluate the effectiveness of the relevant event retrieval process. We construct queries from all of the 461 events in our dataset with different number of events. We first randomly select 46 events as length-1 queries. Then 46 length-2 events are randomly selected by combining any two of the length-1 queries. Next, 46 length-3 events are randomly selected by combining any three of the length-1 queries. After this process, we have 138 queries in total. For each query, all of the returned results are assessed as either relevant or irrelevant; there are 42,733 labeled judgment pairs in total. Standard information retrieval evaluation metrics [72], including precision@3, precision@5, precision@10, Mean Reciprocal Rank (mRR) and Mean Average Precision (mAP), are used to evaluate the results. The first three precision based metrics are considered good metrics for results returned from mobile devices or Web searches. mRR measures the rank of the first relevant result, and mAP considers recall as well as precision in the measurement.

For each query, we divide the query q into separate events. Each potential result i has a relevance score $S(i, j)$ according to Equation 3.6 given event j. Therefore, the relevance score RS_i is computed according to:

$$RS_i = \frac{\sum_j S(i, j)}{|q|} \tag{3.17}$$

The first three rows of Table 3.12 show three sets of contextual retrieval results based on length-1-to-3 queries. The last three rows show three sets of randomized retrieval results based on length-1-to-3 queries.

	P@3	P@5	P@10	mRR	mAP
Q1	0.717	0.625	0.555	0.889	0.691
Q2	0.725	0.675	0.586	0.855	0.668
Q3	0.759	0.722	0.623	0.856	0.683
Q1-Ran	0.017	0.010	0.013	0.057	0.028
Q2-Ran	0.012	0.011	0.012	0.064	0.029
Q3-Ran	0.021	0.019	0.019	0.079	0.032

Table 3.12: Event retrieval ranking results.

As we can see from Table 3.12, using event context, the retrieved results have very good top-K accuracy and very high performance for first relevant result retrieval. By considering recall, the mAP also shows high performance. We also see that queries with different length show very similar performance. This means contextual retrieval is consistent for queries of different lengths. Compared to randomized retrieval methods shown in the last three rows, contextual retrieval can return events that are much more relevant.

The results above show that the semantic relevance of events as labeled by a human. However, there are some cases where, although human may think two events are semantically unrelated, they share the same context. These cases show interesting results for relevant events that can not be captured by semantics. One good example that we found in our dataset is the *2013 Obama inauguration* event, which happened from Jan 19 to Jan 21, 2013 and the *The 2013 International Consumer Electronics Show (CES)* event happened from Jan 8 to Jan 11, 2013. At a first glance by human, these two events are completely unrelated. However, our model shows that these two events have a common context of people who like technology. The discovery of non-semantic but contextually relevant events is the unique benefit of event context identification.

3.4.5 Friendship and Event Context

To emphasize the importance of identified event context, we will show another possible application. In this experiment, we investigate the correlation of identified event context with user friendship. We will demonstrate that users who belong to similar event context are more likely to be friends. This observation allows us to build better friendship recommendation services.

We randomly sample 100,000 users and obtain all of their friends. For each user and friend pair,

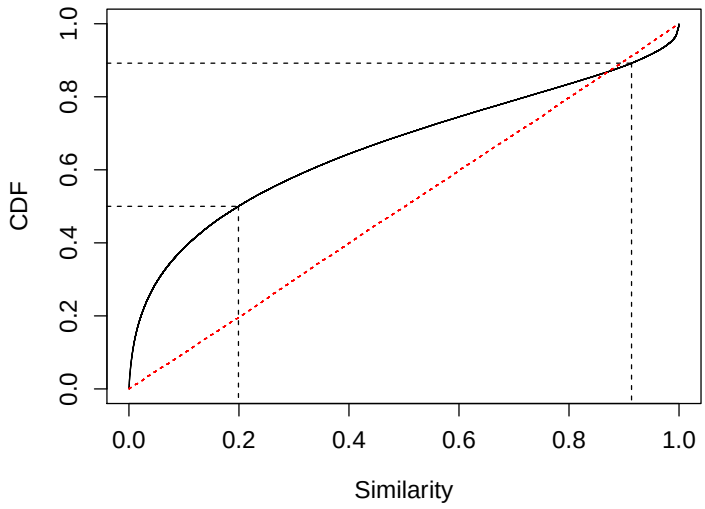

Figure 3.7: CDF of user-friend similarity based on event context.

we calculate the user-friend similarity of their latent factor vectors based on the similarity defined in Equation 3.6 that are generated from the matrix factorization. Let $Rank_f$ be the rank of a friend based on similarity, the relative similarity is calculated for each user as:

$$Rank_f/\#friends \tag{3.18}$$

The reason for defining the relative similarity is that the rank is important, and we want to normalize the rank. A similarity value of 1 indicates the most dissimilar user-friend pair.

We plot the CDF of all the user-friend similarity pairs in Figure 4.9. As we can see from the solid line in Figure 4.9, users tend to have many similar friends. 50% of the friends are within 0.2 similarity. The dotted line shows the case where user-friend similarity is randomly calculated. At the end of the CDF curve, we do see there is a trend of increasing dissimilarity. This means users also have many dissimilar friends. This can affect performance when we consider the use of friends' interests to help predict users' interests; we will verify this further in the next experiment.

The above analysis shows only aggregated similarity between users and their friends. In our second experiment, we want to see the friendship similarity distribution for each user. In Figure 3.8, the x-axis

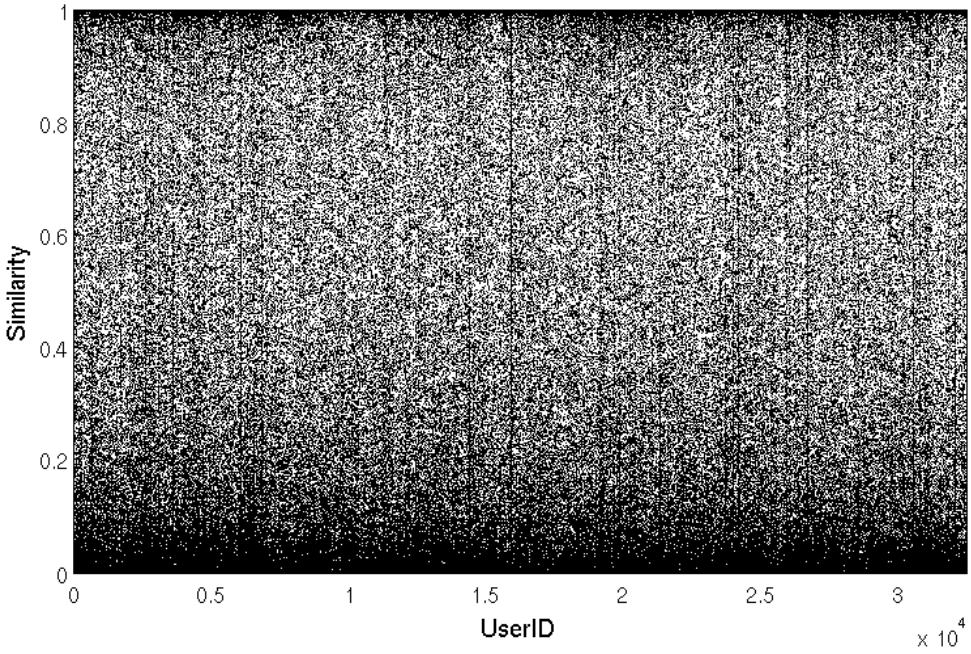

Figure 3.8: Friend similarity distribution per user.

represents users, the y-axis indicates relative friendship similarity as defined above, and each dot represents a friend. As we can see from the figure, there is high density around similarity of 1 and similarity of 0, meaning that users have both similar and dissimilar friends. The dissimilar friends are likely to be loosely-connected social friends and they may not share common interests with the user. This also explains why a direct average of friends' interests, as performed in the SocialMF model, will not help infer users' interests. In reviewing all of the results from the analysis in this subsection, we see that better friendship recommendation can be made based on event context information.

3.5 Conclusions

In this work we have presented AnchorMF, a matrix factorization technique to solve the event context identification problem. AnchorMF selects anchors from users' followers and incorporates anchor information into an extended PMF framework. We have also presented several applications of using identified event context to predict users' participation in events, retrieve relevant events, and recommend friends based on

event context. Our evaluation using real-world Twitter data shows that AnchorMF outperforms existing matrix factorization techniques by 20.0%. In our future work, we would like to explore other potential features, such as location information in users' profiles and tag information from users' followers, and consider how these features can be used in our model for better event context identification.

Chapter 4

Location Inference

4.1 Introduction

With the broad support of anywhere, anytime social interaction in today's online social networks (OSNs), location sharing is becoming prevalent. As another dimension of social networking, users can geotag their posts, and announce their current locations to friends and the public at large. The benefits of location sharing can be multi-fold. It facilitates more effective social networking, as well as producing better location-based intelligence, which is useful for both business entities and end users. Studies have shown that location sharing is useful for many social applications and recommendation services [7, 21, 78].

However, the disclosure of location information raises serious privacy concerns [18, 39, 1, 42]. Long-term location trail can reveal a person's location routines [1]. One key challenge is to determine who has access to a user's location information. This is particularly difficult in OSNs, where the sharing of various types of context information makes it possible for privacy attackers to identify a user's whereabouts. Even when no explicit location information is disclosed by a user, the user's location can be inferred from his/her status updates and also friendship information.

To identify an OSN user's geographic location[1] , previous works have studied the use of either text postings or social structure information [23, 15, 75, 6], and their estimation errors range from 700km–1000km for average error and 140km–500km for median error. In this work, our goal is to investigate new techniques that leverage both textual and social structural information, and understand to what extent users

[1] If a user is associated with multiple GPS coordinates, the centroid (i.e., mean) of the coordinates is used as the user's groundtruth location. We assume that the centroid of a user's location is relatively fixed while occasional travels might happen.

can be geolocated when multiple types of data are fused together. This is a challenging problem due to the following reasons:

- Although large amounts of data exist in OSNs, there are no strong or explicit geographic clues. It is unclear how text postings and social structures can be used for stable and accurate location estimation.
- Information in OSNs is very noisy. User-generated data are diverse in style and content, and there can be various types of errors and/or inconsistencies.
- While geographic clues may be obtained from multiple types of features, their relative importance or confidence are not clear, nor is the fusion strategy.

To address these challenges, we have developed **GeoFind**, a unified solution for identifying users' geographic locations by utilizing both textual and social structural information in OSNs. GeoFind ranks candidate geographic regions (clusters) in a three-stage ranking process: (1) text-based ranking using a combination of geo-sensitive textual features; (2) structure-based ranking using maximum likelihood estimation (MLE) of geotagged friendship information; and (3) fusion (or re-ranking) of text-based and structure-based orderings of geographic regions.

To the best of our knowledge, this is the first work that aims to geolocate users in OSNs through the fusion of text and social structure. This work makes the following contributions:

- Analysis and combination of geo-sensitive textual features for improved text-based location estimation;
- Analysis of friendship versus geographic distance and maximum likelihood estimation for improved structure-based location estimation; and
- Effective fusion (re-ranking) of both text-based and structure-based rankings for accurate location estimation.

We evaluated GeoFind using a public dataset and our own dataset containing 0.8 million geotagged Twitter users collected through a 3-month period. Evaluation results demonstrate that GeoFind outperforms state-of-the-art techniques with significant reduction in estimation error (25% reduction in average error, and 66% reduction in median error). In particular, our solution reduces the median error from 140km–500km to 47km. We further discuss the implications of location privacy given the improved location accuracy through the fusion of multiple data types in OSNs and beyond.

4.2 Problem Formulation and System Overview

In this work, we aim to geolocate users by leveraging both text and social structure information in OSNs. Using Twitter as our test platform, we first collect a large number of geotagged tweets and cluster them into **geographic regions**. The details of the data collection process is described in Section 5.6. Given a test user, our goal is to correctly retrieve one of the geographic regions which has the **maximum similarity** to the actual location of the test user. A formal definition of the location identification problem is given as follows:

Location Identification Problem: Let $\mathcal{X}$ be a list of geographic regions and

$$\mathcal{X} = \{\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3, \cdots\},$$

where $\mathcal{X}_i$ refers to one of the geographic regions. For a given test user

$$u = (T_u, G_u, S_u),$$

where T_u represents textual features, G_u represents friendship features, and S_u is the groundtruth GPS location. Our goal is to find a ranked list $R(u)$, with

$$R(u) = \{\mathcal{X}_{r1}, \mathcal{X}_{r2}, \cdots\},$$

where

$$\mathcal{X}_{r1} = \underset{i}{\operatorname{argmax}}\, similarity(\mathcal{X}_i, S_u).$$

To solve this problem, we firstly use the test user' tweets as a query and obtain a **textual ranking** of all geographic regions according to their textual relevance to the query. Secondly, we use the bidirectional

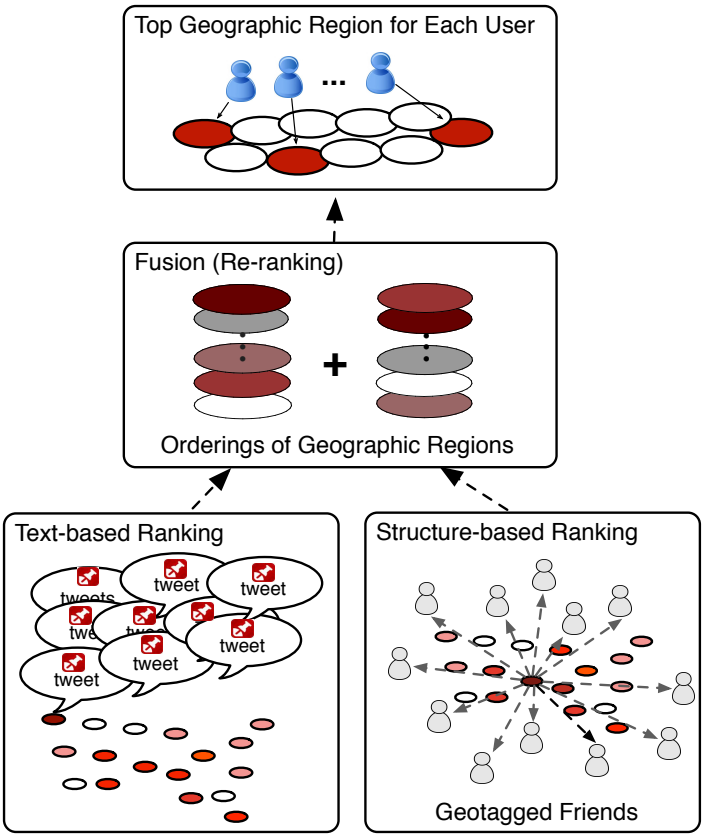

Figure 4.1: GeoFind: Fusing textual and social structure information in online social networks to geolocate users.

friendship links of the test user to obtain a **friendship ranking** of all geographic regions based on their likelihood of containing the test user's location. Finally, we obtain a **re-ranking** of the geographic regions by boosting from both ranked results. We define the key concepts as follows:

- **Geographic region** $\mathcal{X}_i$: Given geotagged tweets $M = \{(m_1, S_1), (m_2, S_2), \cdots\}$, geographic regions $\mathcal{X}_i$ are the result of clustering M based on GPS locations, i.e., each $\mathcal{X}_i$ represents a spatial cluster of M.

- **Similarity**$(\mathcal{X}_i, S_u)$**:** The Earth distance between the average GPS of $\mathcal{X}_i$, and user u's groundtruth GPS location S_u.

- **Textual ranking** R_{T_u}**:** ranking of geographic regions for a given user u based on textual features.

- **Friendship ranking R_{G_u}:** ranking of geographic regions for a given user u based on friendship features.

- **Re-ranking R_{F_u}:** ranking of geographic regions after fusing R_{T_u} and R_{G_u}.

4.2.1 Geographic Regions

Previous works focused mainly on building direct relation from tweet text to specific GPS coordinate. Eisenstein et al. proposed an innovative geographic topic modeling technique to estimate location from a Gaussian distribution [23]. Cheng et al. leveraged an MLE approach to identify location-specific terms and then identify accurate GPS based on the combination of location-specific terms [15]. Wing et al. proposed to cut the Earth surface into grids and map the test user's tweets to one of the grids and the GPS estimation is the center of each grid [75].

We expand the idea of text-to-GPS coordinate mapping and map text to a region on Earth. Note that the regions do not correspond directly to geographic areas such as counties and states, since tweets density is different in each geographic area, known as the sparse data problem. The benefits of mapping text to a region instead of GPS coordinate are multi-fold. First, it is more intuitive to think someone is from a certain region rather than a single point. Second, we can ensure that we have enough data within each region by controlling the size of that region. Third, it is then easier to transform identifying people's locations to retrieving the most likely region for a given person. Fourth, region becomes the most basic element throughout our design, simplifying our design yet still maintaining high accuracy.

Although various clustering techniques have been developed, we choose k-means clustering for both effectiveness and efficiency reasons. This simple and intuitive clustering technique serves the purpose of dividing the map into geographic regions according to all the geotagged data. As we can see from Figure 4.3, k-means successfully divides US into reasonable regions. Here we set k to 200 empirically. The tradeoff of setting different k values is discussed in Section 4.6.2.

4.3 Text-based Location Ranking

Everyday, a huge amount of text data are generated in OSNs, e.g., over 200 million tweets per day on Twitter. Even though only around 1% of the tweets are geotagged, we still have a large amount of text data for location identification. The challenge is to identify **geo-sensitive** textual features despite the fact that user inputs could differ significantly in content and style of writing. Specifically, we would like to select textual features which correlate well with geographic locations, such as the mentioning of local restaurants or sports teams. One other interesting fact we have observed in our textual analysis is that local users are also mentioned in tweets, which motivated us to utilize the social structural information of friendship links (discussed in detail in Section 4.4).

4.3.1 Relevance Ranking Based on Textual Features

Given the tweets of a test user, our goal is to rank geographic regions (each consists of geotagged tweets in that region) based on textual and geographic similarity. Following the traditional bag-of-words model, we first **tokenize** the tweets using the state-of-art TweetMotif tokenizer [17], which is specifically designed for tweets. Also, a Twitter part-of-speech (POS) tagger [30] is utilized for mapping each token to one of the POS tags. The main purpose of this step, as shown later, is to identify the effectiveness of each POS categories in terms of geolocating users. As a design consideration, no stemming is used since we do not want to destroy any possible proper nouns. In order to extract distinctive and geo-sensitive features for each region, we use TF-IDF weighting [63], a well-known textual feature extraction scheme which uses both local and global information to extract feature vectors.

After we have extracted feature vectors for all the geographic regions and test users, we calculate the relevance of regions for each test user. The similarity metric we use is cosine similarity, which measures the cosine of the angle between two feature vectors. The intuition is that, the more textually similar between a test user and a region, the more likely the user belongs to that region. Based on the similarity scores, a ranking of all geographic regions is obtained for each test user, with the first region being most textually similar to the user's tweets.

Table 4.1: Combination of Different Textual Feature

FeatureSet #	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
common noun	all^*	✓	X	✓	✓	✓	✓	✓
proper noun		✓	✓	X	✓	✓	✓	✓
proper noun & possessive		✓	✓	✓	X	✓	✓	✓
nominal & verbal		✓	✓	✓	✓	X	✓	✓
hashtag		✓	✓	✓	✓	✓	X	✓
at-mention		✓	✓	✓	✓	✓	✓	X
median(km)	168	130	164	261	129	126	122	535

All textual tokens are used, not just the POS features listed in this table.

4.3.2 Geo-sensitive Textual Features

Although textural features are generally believed to be useful for location identification, it is unclear which textual features are geo-sensitive and how useful they are. According to [30], there are potentially 25 part-of-speech (POS) features on Twitter. Many of them are obviously not correlated to geographic locations, such as verbs, determiners(the, its, her), etc. Others are possibly geo-sensitive, e.g., proper noun(Alcatraz, Lebron, NY), common noun, etc. In order to analyze the POS features effectively, we first focus on a subset of key features which we believe are possibly geo-sensitive. By removing other features, we create the baseline results which should not be worse than keeping all the features. Next, among the key features, we remove each feature and use the remaining features as a combination feature set for location identification, i.e, leave-one-out. If the results are significantly worse than the baseline, it means that the feature removed is important and geo-sensitive.

As shown in Table 4.1, set #1 contains all the tokens we have extracted from each test user's tweets, and set #2 contains all the key POS features. For set #3 to set #8, we remove one key feature each time. The bottom row shows the median distance error (see definition in Section 4.6.1) of all users' location estimations, i.e., 50% of the test users are located within a certain distance from their true locations. We see that set #2's result is actually slightly better than that of set #1 (130km vs. 168 km), which indicates that the key features we select are appropriate. Compared to the baseline result (set #2), set #3, set #4, and set #8 perform worse (164km, 261km, and 535km, respectively). The results indicate that, among all the POS features, **common noun**, **proper noun** and **at-mention** are more geo-sensitive. The first two features

are intuitive, as we can imagine the names of local sports teams or restaurants appear in the two features. It is actually surprising to see **at-mention** phrases play a significant role in location identification. Table 4.1 shows that for set #8, feature combination without at-mention dramatically decreases the performance by about 4 times. At-mentions represent conversations between users, which usually involve local users in the same geographic region. This effect should be further explored to see who are following the local users and the friendship structures.

4.4 Structure-based Location Ranking

Based on our analysis of geo-sensitive textual features, we found that many Twitter users' names appear in geotagged tweets, and the mentioning of these names reflect significant geographic relevance. The intuition is that Twitter users in a given geographic region is likely to follow other local users, which may be beneficial for estimating users' locations. In traditional OSNs such as Facebook, it has been observed that the social network structure is to some extent correlated with users' geographic distribution [6]. Twitter, however, exhibits some non-social properties similar to a news media [43]. It is not clear if and to what extent Twitter users' social structure correlates with their geographic locations. Furthermore, Twitter friendship links are directional: user A following user B does not necessarily mean that user B follows user A, which is different from Facebook's friendship links.

Given the characteristics described above, we propose a structure-based location ranking method which differs from FindMe [6] in the following important aspects. First, our method positions each user based on the assumption that more of his/her friends are likely to be close by and fewer friends are further away, i.e., smaller the distance, larger the fraction of friends in that distance. FindMe assumes a user should be close to his/her friends as well as being further away from non-friends, which may be too restrictive given that people in the same region are not necessarily friends and real-world friends in the same location are not necessarily reflected in OSNs. Second, FindMe considers both friends and non-friends (i.e., all users in an OSN) for each user, which has a high computation overhead and do not scale. Our method, on the other hand, considers only friends for each user, which is much more efficient and scale well with the size of the OSN. Finally, FindMe uses a linear regression approach to estimate its parameters, while we propose an

MLE-based approach for parameter estimation with much better accuracy. Next, we explain in detail our MLE-based ranking framework, friendship modeling and parameter estimation.

4.4.1 MLE-based Ranking Framework

Given the observations of friendship links in an OSN and some geotagged users, our goal is to determine the geographic location of a given user. Let $\mathcal{L}(u, s)$ be the likelihood of user u being located at location s, G_u be the set of geotagged friends of u, and s_{g_i} is the location of each friend $g_i \in G_u$. Assuming independence of the friends' locations, we define

$$\mathcal{L}(u, s) \propto \prod_{g_i \in G_u} \mathbb{P}(u, s|g_i, s_{g_i}), \tag{4.1}$$

where $\mathbb{P}(u, s|g_i, s_{g_i})$ denotes the probability of u at location s given that a friend g_i is located at s_{g_i}. To avoid the underflow problem, we transform the equation as follows:

$$\ln \mathcal{L}(u, s) \propto \ln \sum_{g_i \in G_u} \mathbb{P}(u, s|g_i, s_{g_i}) \tag{4.2}$$

To simplify the computation, we consider the **distance** between u and each friend g_i instead of their exact locations. The assumption is that the distance between u and his/her friends follows a certain distribution and friends are more likely to occur at shorter distances from u. Therefore, the task of calculating $\mathbb{P}(u, s|g_i, s_{g_i})$ can be replaced with the estimation of $p(d)$, the probability of u having a friend who is distance d away from u. This is achieved through friendship modeling and parameter estimation, described in the following subsection. For each of the candidate geographic regions (Section 4.2.1), we calculate the corresponding likelihood of user u being located at the center of that region by calculating the distance d between u and each friend and summing up the corresponding $p(d)$ values. All the geographic regions are then ranked in descending order for user u based on the likelihood values.

4.4.2 Friendship Modeling and Parameter Estimation

Our first step is to model the friendship-distance relationship using a function that takes the distance as input and outputs the probability of friendship between users. In other words, the function captures the

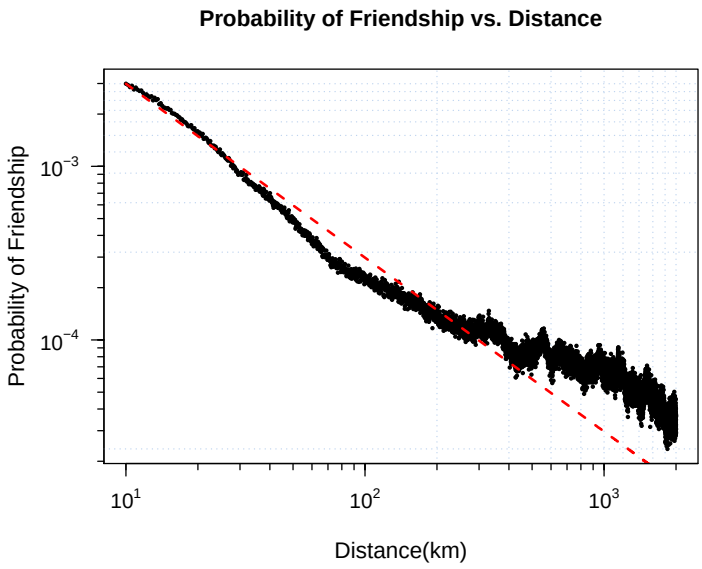

Figure 4.2: Probability of Friendship vs. Distance in log-log scale. Dotted line is the fitted power-law distribution.

probability of somebody's friends occurring at a certain distance. Using our 0.8 million Twitter user data set (Section 4.6.1, we calculate all the pairwise friendship distances (excluding the test users), and split the range of distance evenly into 10,000 buckets between the minimum and maximum distance in the data set. Next, we assign each distance to the corresponding bucket, count the number of friendship distances in each bucket and divide the count by the total number of friendship distances as the probability of friendship for that specific distance bucket. After plotting the probability of friendship as a function of distance in log-log scale, it turns out that the line is a good fit to power-law distribution, as shown in Figure 4.2.

Following this observation, we fit our empirical friendship-distance data into a power-law model:

$$p(d) = Cd^{-\alpha} \quad \text{for } d \geq d_{min}, \tag{4.3}$$

where C, d_{min}, and α are parameters whose values we will determine later. Since we are modeling the probabilities of friends occurring at certain distances, the sum of the probability values should equal to 1 and the power-law function needs to be normalized before we estimate the parameters. Let $\int_{d_{min}}^{\infty} p(d) = 1$, we can derive that $C = (\alpha - 1)\, d_{min}^{\alpha-1}$. Note that the derivation is only meaningful with $\alpha > 1$, which is a

requirement for a power-law function to normalize. Thus,

$$p(d) = \frac{\alpha - 1}{d_{min}}(\frac{d}{d_{min}})^{-\alpha} \quad \text{for } d \geq d_{min} \tag{4.4}$$

Now that we have the normalized power-law model, we are ready to estimate parameters d_{min}, and α. Here, d_{min} measures the minimum friendship distance we take into consideration. Mathematically, we can use the minimum distance in our data set (which is very small and close to zero) as d_{min}, but in reality, it would give us too much noise for extremely close GPS coordinates. On the other hand, if we choose a big d_{min}, we may end up cutting off too many nearby GPS points that may be useful for location prediction, resulting in a biased data sample and eventually poor estimation results. Hence, we want d_{min} to be small enough to have enough precision but not too small to bring in much noise. Based on this reasoning, we only consider distance values that are at least 10km. α represents the power of friendship: the smaller α is, the more friendship links there are. Previous research in social networks shows that α is typically close to 1 [22]. We now estimate the parameters from the empirical data using the goodness-of-fit test based on Kolmogorov-Smirnov statistics and maximum-likelihood fitting methods [16].

Estimating d_{min}: In general people use MLE for parameter estimation, but this approach is not directly applicable to the estimation of d_{min} because increasing d_{min} would actually decrease the number of d values observed in our data set and thus the likelihood function will never decrease. One possible approach is to choose d_{min} based on KS statistic. Let $\mathcal{D}$ be the set of friendship distances in the data set that is at least d_min, the KS statistic is defined as below:

$$D = max \mid P(d|\hat{\theta}) - S(d) \mid \quad \text{for} \quad d \in \mathcal{D}, \tag{4.5}$$

where $P(d|\hat{\theta})$ is the theoretical CDF with parameter θ, and S(d) is the empirical CDF. The statistic measures the largest deviation in cumulative density between the empirical data and fitted model. Minimizing KS statistic allows us to find the d_{min} that has the closest fitted model to the empirical data. Each time we pick a value for d_{min}, we have to resize the dataset, re-estimate α based on the new setting, and calculate the gap between theoretical and empirical CDFs. After testing on different d_{min} values, we then choose the d_{min} that gives the smallest KS statistic.

Estimating α: Now that we have the estimated d_{min}, we can use the method of maximum likelihood to derive an MLE for α. The derivation process is shown as below:

$$\ln \mathcal{L}(\mathcal{D}|\alpha) = \ln \sum_{d_i \in \mathcal{D}} \frac{\alpha - 1}{d_{min}} (\frac{d_i}{d_{min}})^{-\alpha} \tag{4.6}$$

$$= n \ln \frac{\alpha - 1}{d_{min}} - \alpha \sum_{d_i \in \mathcal{D}} (\frac{d_i}{d_{min}}) \tag{4.7}$$

Let $\partial\mathcal{L}/\partial\alpha$ = 0 and we have:

$$\alpha = 1 + n / \sum_{d_i \in \mathcal{D}} \ln(\frac{d_i}{d_{min}}) \tag{4.8}$$

4.5 Fusion (Re-ranking)

Two different orderings of geographic regions are obtained from (1) text-based ranking using a combination of geo-sensitive textual features and (2) structure-based ranking using MLE of geotagged friendship information. If we view the text- and structure- based rankings as the outputs of two basic models, the challenge is to determine when to put more trust on one model versus the other. While in some cases it is fine to choose one out of the two, in most cases, the combined results of both models would achieve the best performance. Therefore, to take advantage of both text- and structure-based models, we propose a re-ranking algorithm under the learn-to-rank framework [48, 24, 9] to effectively fuse together the two ordered lists of geographic regions.

4.5.1 Properties of Our Re-ranking Problem

Fusing the rankings of two different models to derive a new model is a non-trivial task. Although specific combination rules may be defined, it is difficult to determine all the parameters and has the risk of overfitting. Data in OSNs are diverse and change rapidly, and it is important for us to find an effective way to automatically combine multiple evidences. The learn-to-rank framework is suitable in this case. Essentially, results generated from previous stages can be treated as features to build a learn-to-rank model. Features generated from test data can then be combined automatically and effectively. Here, we highlight some important properties of our re-ranking problem:

(1) Since we use the average GPS coordinates of a geographic region to predict a user's location, we only care about the top-ranked region in the final result. This means we want the top one result to be the relevant result and all others irrelevant.

(2) For the evaluation step of learn-to-rank, it is only positive when the relevant result is the top one result and the ordering of all other regions is not important.

(3) Relative order is important and there is no need to generate the actual score or value accurately.

However, the properties we identify above do not match the assumptions of existing learn-to-rank algorithms. Pointwise approaches do not consider relative order between geographic regions and only predict the relevance degree, which contradicts property 3. Pairwise approaches create highly imbalanced training set based on property 1, since there is only one relevant region in our setting. Listwise approaches rely on permutation-based evaluation, which is not applicable according to property 2.

4.5.2 Re-ranking Algorithm Design

Based on the analysis above, we design our re-ranking algorithm as follows. For a given user, each geographic region is represented by a four-dimensional region feature vector consisting of text-based rank, textual relevance value, structure-based rank, and the corresponding log-likelihood value. Given two different regions, we compute the difference between their region feature vectors and try to determine if the first region is more likely to be the true region than the second one. Our goal is then to map the region difference vector to the 1 or 0 decision. We accomplish this in two stages. In Algorithm 2, we generate two sets of training data (with 1 or 0 labels) based on the training users' true region and its difference from all other regions.

In Algorithm 3, we first use the training data to learn the parameter $\vec{\theta}$ for the logistic function $f(z)$. Since the logistic function is monotonic, for each test user, we examine each region $\mathcal{X}_i = \vec{x}_i$, and the region with the maximum $z_i = \vec{\theta} \cdot \vec{x}_i$ is selected as the test user's relevant region $\mathcal{X}_{max} = \vec{x}_{max}$. Note that pairwise region comparison is not needed here, since for any other region $\mathcal{X}_i = \vec{x}_i$, $z_{max} - z_i = \vec{\theta} \cdot \vec{x}_{max} - \vec{\theta} \cdot \vec{x}_i = \vec{\theta} \cdot (\vec{x}_{max} - \vec{x}_i) \geq 0$, and $f(\vec{\theta} \cdot (\vec{x}_{max} - \vec{x}_i)) \geq 0.5$, so the label would be 1, meaning that region $\mathcal{X}_{max}$ ranks

Algorithm 2 TrainingData()

Input: Set of training users$\{u_i\}$, each user has a list of possible regions $\{\mathcal{X}_i|\mathcal{X}_i = \vec{x_i}\}$, where $\mathcal{X}_{r1}$ is the user's the true (relevant) region, $\mathcal{X}_{r2}$ to $\mathcal{X}_{rk}$ are irrelevant regions, and $\vec{x_i}$ is the region feature vector.
Output: Two sets of training data, $\{v_i|v_i = (\vec{y_{vi}}, l_{vi})\}$, $\{w_i|w_i = (\vec{y_{wi}}, l_{wi})\}$, where $\vec{y}$ is the region (difference) feature vector and l is the label.
for each training user u_i **do**
  for each region $\mathcal{X}_i (i \neq r1)$ **do**
    $\vec{y}_{vi} = \vec{x}_{r1} - \vec{x}_i$; $l_{vi} = 1$
    $\vec{y}_{wi} = \vec{x}_i - \vec{x}_{r1}$; $l_{wi} = 0$
  end for
end for
return $\{v_i\}, \{w_i\}$

Algorithm 3 TrainingAndTesting()

Input: Two sets of training data, $\{v_i|v_i = (\vec{y}_{vi}, l_{vi})\}$, $\{w_i|w_i = (\vec{y}_{wi}, l_{wi})\}$, set of test users$\{u_j\}$.
Output: Relevant region for each of the test users $\mathcal{X}_{max}$
$\vec{\theta} = LogisticRegression(\{v_i, w_i\})$
for each test user u_j **do**
  for each region $\mathcal{X}_i = \vec{x}_i$ **do**
    $z_i = \vec{\theta} \cdot \vec{x}_i$
  end for
  $\mathcal{X}_{max}[u_j] = \mathcal{X}_i$ with the maximum z_i
end for
return $\mathcal{X}_{max}$

higher than any other region $\mathcal{X}_i$.

Algorithm 3 is designed to learn a function, and given a pair of geographic regions, it determines which region ranks higher. However, an arbitrary classifier might predict region $j >$ region i, region $i >$ region k, and region $k >$ region j, for $i \neq j \neq k$. Also if region a ranks higher than region b, the classifier might also predict region v ranks higher than region u. In these cases we cannot get the highest ranked region without conflict. Our algorithm guarantees that there is no conflict case. We can easily verify the following theorem according to the monotonic property of logistic function:

Theorem 1. *Let f be the logistic function learned by the algorithm. If $f(a-b)=1, f(b-c)=1$, then*

$$f(a-c)=1$$

If $f(u-v)=1$, then

$$f(v-u)=0$$

Proof. Since f is a logistic function, we know

$$f(z)=\frac{1}{1+e^{-z}}, z=\beta_0+\sum_{i=1}^{n}\beta_i x_i$$

So

$$z(a-b)=z(a)-z(b)>0, z(b-c)=z(b)-z(c)>0$$

And

$$z(u-v)=z(u)-z(v)>0$$

Then

$$z(a-c)=z(a)-z(c)>0, z(v-u)=z(v)-z(u)<0$$

which shows that

$$f(a-c)=1, f(v-u)=0$$

□

4.6 Experimental Evaluation

In this section, we evaluate to what extent individual users can be geolocated using their textural postings and social structure information in OSNs. Specifically, in comparison with existing state-of-the-art methods, we want to evaluate how GeoFind perform overall and how the individual components perform, including GeoFind text-only, structure-only, and fusion (re-ranking).

4.6.1 Experimental Setup

Two different data sets are used in our experiments. The first data set is from the authors of [23]. It contains one-week geotagged tweets within the U.S. from the Twitter Stream API. The dataset is also used in [75]. The second data set contains data we collected from Twitter over a 3-month time period from June to September 2011. This data set contains 0.8 million geotagged Twitter users within the U.S., which have been obtained from the Twitter Stream API using a fixed geographic. We also obtain the users' social structure information and restrict the users to have more than 10 and fewer than 1000 friends and followers. Users outside that range do not show consistent social behavior on Twitter [43]. For each user with multiple geotagged tweets, the user's true location is computed as the average GPS coordinates of the tweets we collected from that user. In total, we have 0.82 million users. From these users, we randomly selected 10,000 users and collected all the tweets from these 10,000 users, giving us 16,436,545 tweets in total. Table 4.2 summarizes the key characteristics of these two data sets. For data set 2, all the users are clustered into 200 geographic regions, and their distribution in the U.S. is visually shown in Figure 4.3, where circles with larger diameter represent clusters with more users.

From figure 4.4, we see the distribution again from a different source of data. There are 18 months of all geo-tagged tweets from Twitter firehose, around 2.7 billion in total, provided by Gnip and Mapbox [29], where brighter color (darker in grey color mode) represents places with more users. Compared to figure 4.3, we find the two distributions are very similar to each other. It means our dataset is representative and follow the distribution of real geo-tagged tweets distribution on Twitter.

We follow the same evaluation metrics using in [15, 23, 75]. For each test user $u_i \in U$, we calculate

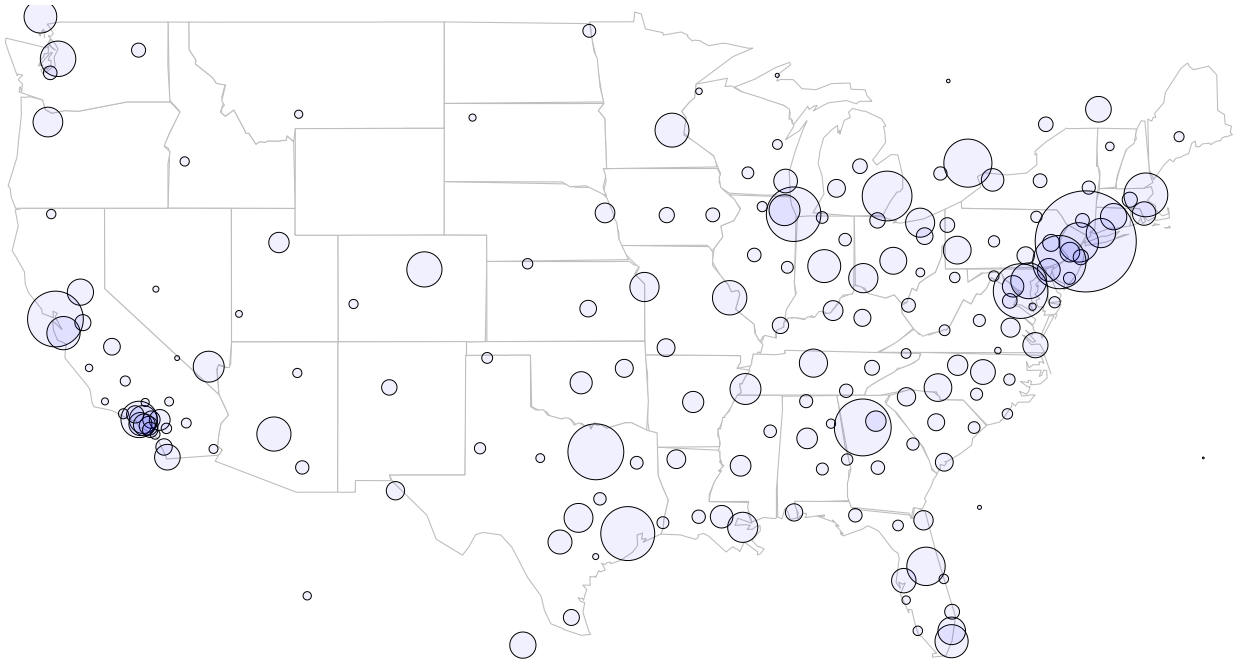

Figure 4.3: Distribution of 200 geographic regions (clusters) obtained from data set 2. Bigger circles represent more users.

the *Error Distance*, $error_i$, which represents the Earth distance between the estimated location, S_{est} and the true location, S_{true}:

$$error_i = EarthDist(S_{est}, S_{true})$$

Given the large number of test users, we define *Average Error Distance* and *Median Error Distance* as follows:

$$Avg_{Err} = \frac{\sum_{i=1}^{|u|} error_i}{|u|}$$

$$Median_{Err} = sorted(error)_{n/2}$$

For our GeoFind solution, we consider three variations: text-only, structure-only, and fused. We compare GeoFind to the following techniques:

- **Text-based techniques**: GeoLex [23] extends LDA to incorporate location information and derive geographically correlated sub-topics. New tweets can then be mapped to the sub-topics for location identification purposes. SimpGeo [75] is a grid-based approach which maps textual features

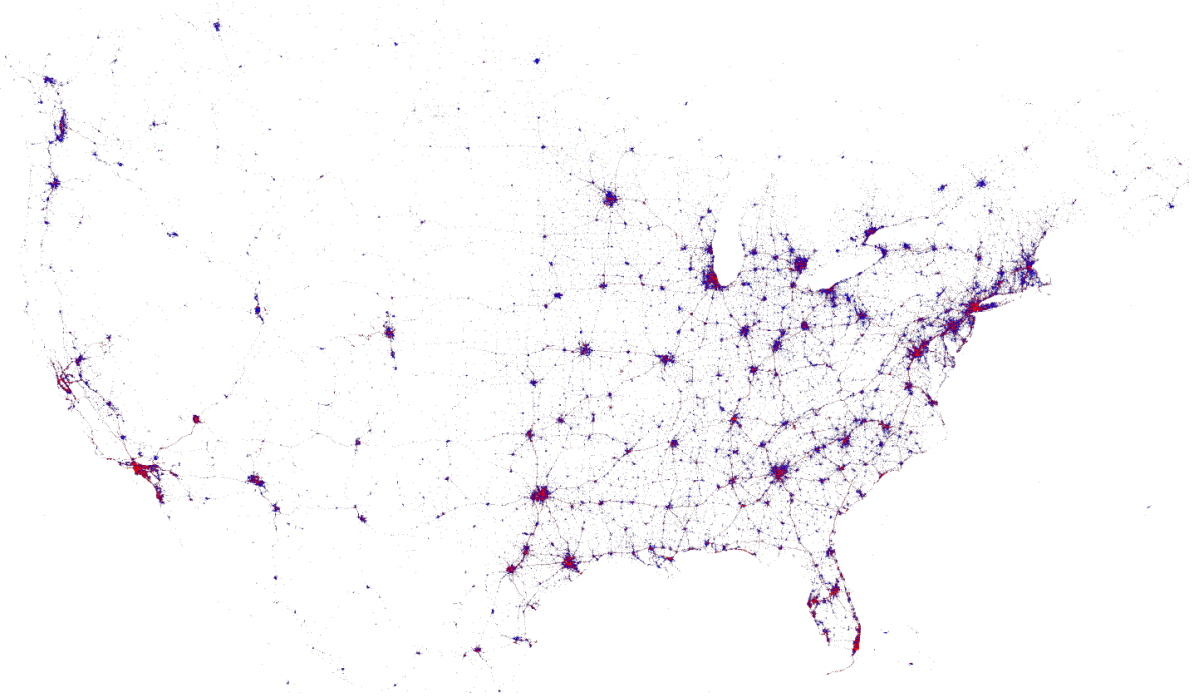

Figure 4.4: Distribution of 18-month of all geo-tagged tweets from Twitter firehose. Brighter color (darker in grey color mode) represents more users.

according to the language model they developed. Note that both techniques used data set 1 for their evaluations, and their reported results are included in Table 4.3. LocalTweet [15] is an MLE-based technique which captures local words for location identification. It was evaluated on a different data set which we do not have access to. The results are also included in Table 4.3.

- **Structure-based technique**: FindMe [6] utilizes both friends and non-friends data to predict location information on Facebook. The key difference between their approach and ours is described in Section 4.4. We have implemented the FindMe method and compared it with GeoFind using data set 2.

- **Learn-to-rank techniques**: We also compare our re-ranking algorithm to RankBoost [24] and RankNet [9]. Both are well-known pairwise learn-to-rank algorithms and are implemented in RankLib [58], a library of ranking algorithms, which we used for the comparison in Section 4.6.5.

All the tweets have been pre-processed in parallel using a 48-core machine, 2.2GHz for each core,

Table 4.2: Data Set Properties

Data set	#Users	#Tweets	#Friendships	Duration	Size
1	9,474	377,616	n/a	1 week	55MB
2	0.82M	16.4M	11.5M	3 months	∼7.5GB

78.2GB memory in total, running Ubuntu 10.04.3, with JVM 1.6.0_26 installed. All other experiments have been conducted on a single core with 2.4GHz processor and 47.3GB memory available, on a machine equipped with 16 cores and running Ubuntu 10.04.3 and JVM 1.6.0_26. All the implementations and experiments are written in Java.

4.6.2 Overall Quality

Table 4.3 compares the overall quality between our GeoFind solution and state-of-the-art techniques. GeoFind significantly outperforms other approaches, achieving 534km average error and 47km median error for data set 2 with 0.8 million users. Compared with FindMe, the next best-performing technique, GeoFind reduces average error by 25% and median error by 66%. And the reductions are even bigger when compared with the reported results of other text-based techniques. In addition, we can make the following observations from the table: (1) GeoFind text-only outperforms other text-based techniques (GeoLex, SimpGeo, and LocalTweet); (2) GeoFind structure-only outperforms other structure-based technique (FindMe); (3) GeoFind structure-only outperforms GeoFind text-only; and (4) GeoFind fusion outperforms text-only and structure-only, and has the best performance overall. The results demonstrate that in real-world OSNs, it is possible

Table 4.3: Overall Quality Comparison

Error Metric (km)		Dataset1 Average	Dataset1 Median	Dataset2 Average	Dataset2 Median
GeoLex		900	494	-	-
SimpGeo		967	479	-	-
LocalTweet		860	≈ 160	-	-
FindMe		-	-	715	140
GeoFind	Text	730	130	782	146
	Structure	-	-	625	50
	Fusion	**730**	**130**	**534**	**47**

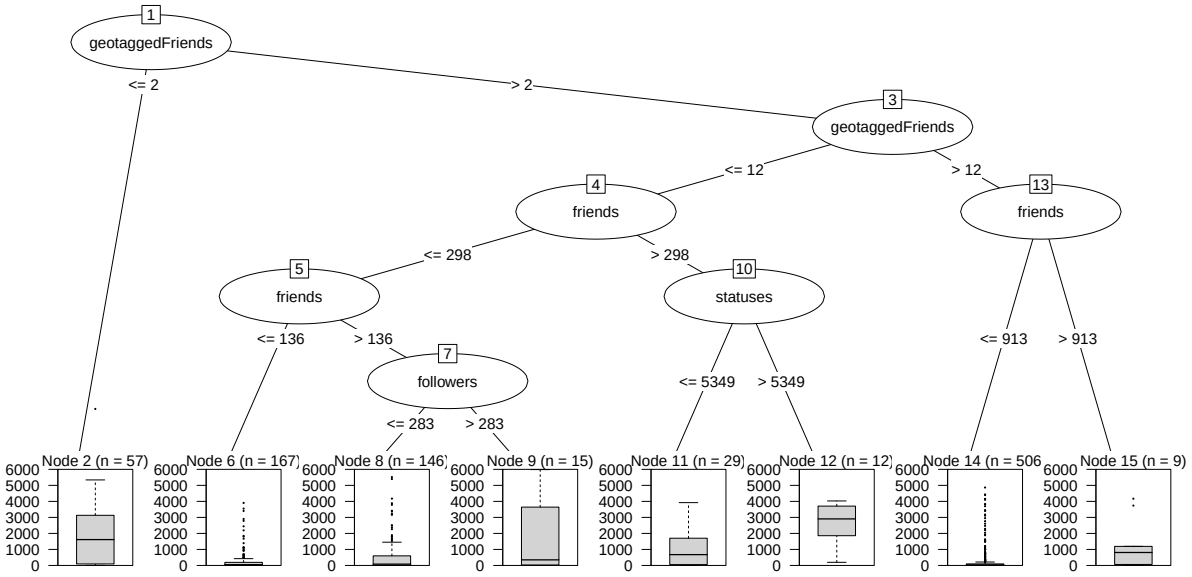

Figure 4.5: Conditional regression tree showing the estimation error under different situations, e.g., Node 14 with high estimation accuracy.

to geolocate users with reasonable accuracy by leveraging both textual and social structural information.

One important question to answer is — under what situations the prediction of GeoFind is accurate. Figure 4.5 shows the conditional regression tree of 941 randomly-selected users based on features including number of geotagged friends, number of friends, number of tweets, etc. Each node contains a subset of the users satisfying specific conditions (e.g., Node 2: at most 2 geotagged friends). The bottom level shows a boxplot of estimation errors for the users within each node. We can see that Node 14 outperforms all other nodes and it represents situations when a user has enough geotagged friends (> 12) but not too many friends in total (≤ 913). For all the test users belonging to Node 14, the average error is 263 km, and the median error is 28 km – almost 50% boost compared with our overall results.

For a further inspection of highly populous areas, we show the results of top 10 most populous counties in the U.S. (Table 4.4). GeoFind performs well for eight out of the ten counties, with median error ranging from 8km to 23km. And the good results are achieved with only small number of samples (i.e., number of training Twitter users in each county).

One important parameter in our design is the number of geographic regions (i.e., clusters) we set in

Table 4.4: Results of Top 10 Most Populous Counties

County	#Samples	AvgError (km)	MedianError (km)
Los Angeles	78	505	16
Cook	42	210	18
Harris	26	484	18
Maricopa	22	929	412
San Diego	24	347	105
Orange	45	281	23
Kings	14	255	8
Miami-Dade	20	674	22
Dallas	23	262	21
Queens	12	46	11

the initial clustering procedure. As shown in 4.6, we experimented with different number of clusters between 50 and 300 with an interval of 50. Most of them performed similarly except for 50, in which case the number of regions is too few to achieve reasonable accuracy. On the other hand, when the number of regions is too many, structure-based ranking is not affected, but text-based ranking would have poor performance since the amount of data is not sufficient in small regions. Empirically, the number of regions can be set to between 150 and 250.

We also observe that the median error goes down as the number of clusters goes up, but the average error goes up as the number of clusters goes up in the end. This is because if the number of clusters is too large, it will never hurt the effectiveness of structural ranking, and it dominates the median error in GeoFind. On the other hand, the the number of clusters is too big, it will hurt the textual ranking performance, since we may not have enough data in too small regions, and that is why the average error will eventually go up.

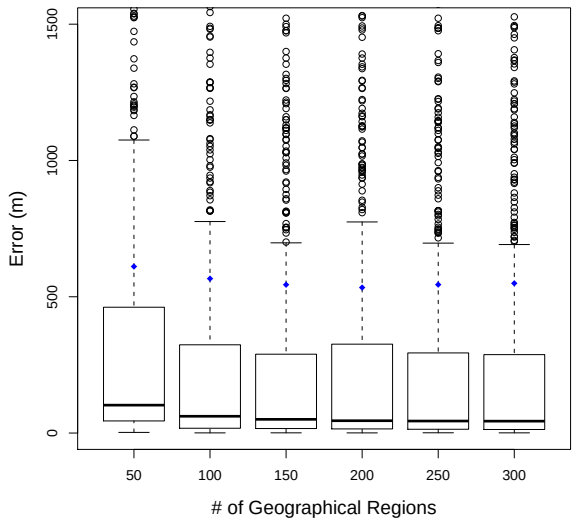

Figure 4.6: Parameter Tuning on Number of Regions

4.6.3 Quality of Text-based Location Ranking

As shown in Table 4.3, for data set 1, which contains only text information, our GeoFind text-only solution outperforms both GeoLex and SimpGeo, as well as the results reported by LocalTwitter on their

own data set. GeoFind text-only also performed similarly for data set 2, with 0.8 million Twitter users and 16.4 million tweets. To further analyze how individual POS features contribute to the overall performance, we consider different feature combinations and their corresponding results. As shown in Table 4.1, feature sets 1, 3, 4, 8 perform worse than others. This shows that it is noisy to use all the tokens we have (set #1). Also, common noun and proper noun play an important role for location identification (set #3 and set #4). This is intuitive since we can imagine local restaurants, sports teams have strong geographic clues. Finally, the most important feature (which we cannot drop) is the at-mention feature (set #8). This shows strong local interactions among local users, i.e., local people tend to follow local information sources and interact with them. This important finding motivated us to include social link structures for location identification.

4.6.4 Quality of Structure-based Location Ranking

In this sub-section, we use data set 2, since data set 1 does not contain any friendship information. We randomly select 10,000 test users from the 0.8 million users and used the remaining ones as training users. For each test user, we find all his/her geotagged friends from the training set. We compare our GeoFind structure-only solution with FindMe, a recent structure-based location estimation mechanism. We compare their performances when different percentage of samples (training users) are available, ranging from 100% to 10% with 10% interval, as well as 5%, 3%, 1% and 0.1% to see the extreme cases. Results for 50%, 1%, and 0.1% are shown in Figure 4.7.

From the results we can see that GeoFind structure-only outperforms FindMe when we have more than 1% of the training set. The reason for the improvement is that GeoFind assumes more friends are close by, while FindMe also restricts that non-friends have to be further away, which may not always be true. However, with only 0.1% or even fewer data samples, FindMe outperforms GeoFind dramatically. The reason is that FindMe utilizes non-friendship information and it helps when there are not enough friends for each user.

We also compare the efficiency between GeoFind structure-only and FindMe. Shown in 4.8, as data size increases, the running time of FindMe shows a slope of 1.37, while GeoFind structure-only shows a slope of only 0.0001, i.e., GeoFind is almost constant while FindMe is linear. The main reason is that

GeoFind considers only the number of friends of a user, which is almost constant, while FindMe needs to consider all the existing users (friends and non-friends), so it is linearly correlated to the volume of the data set. As such, GeoFind is much more efficient and scalable in practical usage. For example, with 0.8 million users, GeoFind takes around 6 seconds while FindMe takes around 160,000 seconds. Both were run on the same single 2.4GHz core with sufficient memory.

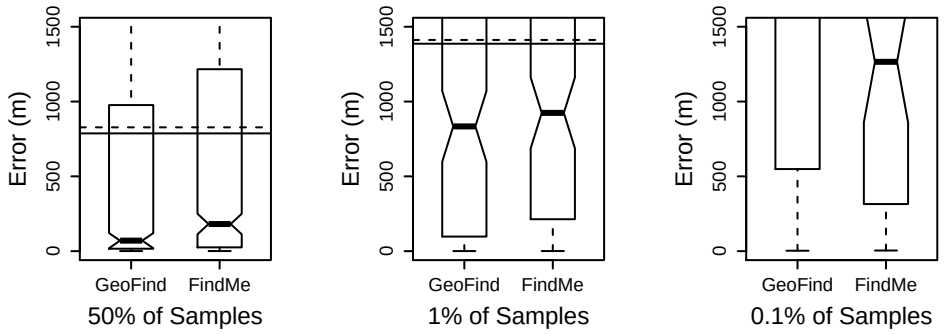

Figure 4.7: Quality comparison of GeoFind structure-only and FindMe.

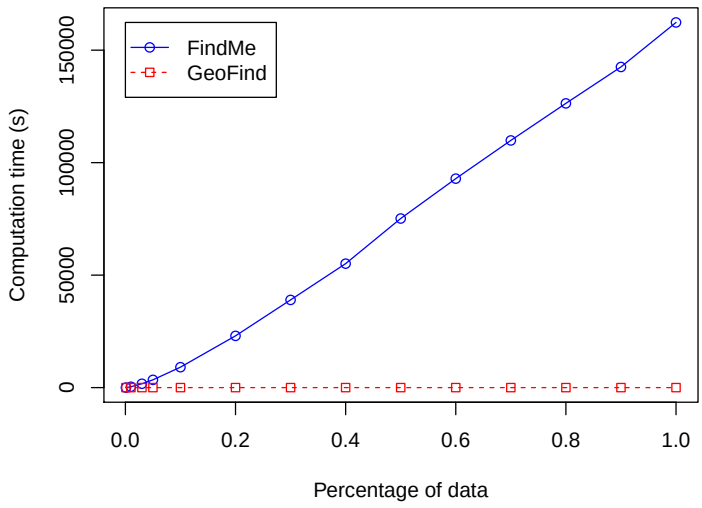

Figure 4.8: Efficiency comparison of GeoFind structure-only and FindMe.

Table 4.5: Re-rank Results

Metric	Avg(km)	Median(km)
RankBoost	1275	915
RankNet	1645	1093
GeoFind	**534**	**47**

4.6.5 Quality of Re-ranking

In this subsection, we evaluate the performance of our re-ranking algorithm and compare with two other learn-to-rank algorithms, RankBoost [24] and RankNet [9].

Since both RankBoost and RankNet are pairwise algorithms, we need to adapt our problem to theirs. We divide the training data by query, which is the test user. Each user has a list of geographical regions, and each region is represented by a feature vector. We label the true region to be relevant and all others irrelevant. The algorithms train all the region pairs according to MAP@1 (mean average precision at top 1 position). Given a test user, the algorithms predict the ranking of all the geographic regions related to this user. Based on our experiments, as shown in 4.5, the average and median estimation errors are 1,275km and 915km for RankBoost, 1,645km and 1,093km for RankNet, respectively. These results are much worse than GeoFind's results of 534km average error and 47km median error. Neither RankBoost nor RankNet performs well, since there are too many irrelevant pairs for them to learn. On the other hand, our re-ranking algorithm is specifically designed to solve this problem. Figure 4.9 shows the boosting results of GeoFind when textual and structural features are fused together. We see that GeoFind benefits from structural features in terms of median error and reduces the average error by considering textual features.

4.7 Discussions

Location sharing and location privacy have been topics of constant debate. As location sharing becomes more prevalent in OSNs, new issues arise. On one hand, OSNs are open and ubiquitous platforms, which are beneficial for location sharing and location based services (LBS). Our GeoFind solution can potentially be used to support automatic location sharing and better LBS. Our results show that in highly populous counties within the U.S., most of the predictions are within a median error of 25km. This is a

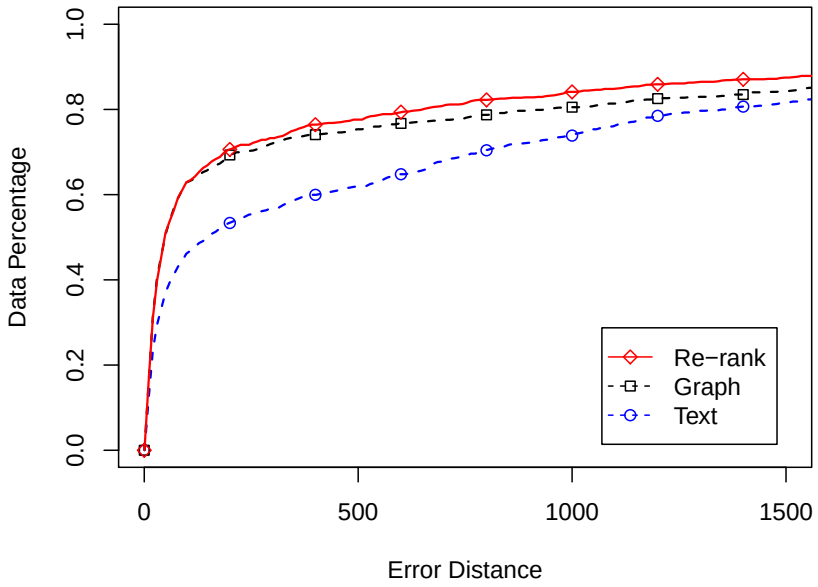

Figure 4.9: GeoFind: Text-only vs. structure-only vs. re-ranking.

quite reachable distance for LBS. Also, third-parties can easily retrieve geographic statistics from OSNs by utilizing GeoFind. This can be beneficial in applications such as crisis management, disease surveillance,etc.

On the other hand, even when an OSN user does not disclose any of his/her location information, it is possible to learn the user's location through the company he/she keeps. Words from users' posts, especially words revealing location features like restaurants, local attractions will disclose the location. Interactions with other users explicitly through at-mentions can potentially disclose more information since other users may be geotagged or geolocated. Our results show that, by fusing textual and social structural information, GeoFind can achieve a median error of 47km for users in the U.S., which is within city range. The fact that the results are comparable to IP-based geolocation (57.2% users within 40km) raises new privacy concerns. Although most people do not disclose their IP addresses in OSNs, the data needed for GeoFind is easily accessible in OSNs, making the privacy concerns more serious. The results of our work call for a re-examination of traditional privacy control policies which tend to only control the content the current user is sharing. New privacy control policies and privacy protection mechanisms may be required.

In this work, we consider the fusion of textual and social structural information in OSNs. There are potentially other new ways to identify private location information that similar to or extend from the techniques we propose in this work. For instance, video/photo data gathered from websites such as Flickr can potentially be leveraged from both content and social structure perspectives. Networking activities, when monitored closely, may also disclose important information make fine-grained location identification possible.

4.8 Conclusions

As location sharing becomes more prevalent in OSNs, we show that even when no explicit location information is disclosed by a user, his/her location can be learned through the company he/she keeps. To demonstrate this, we have designed **GeoFind**, an effective location identification mechanism which leverages both textual and social graph information in OSNs. GeoFind identifies the most likely geographic region for a given user through (1) text-based ranking using geo-sensitive textual features; (2) graph-based ranking using maximum likelihood estimation (MLE) of geotagged friendship information; and (3) fusion (re-ranking) of the text- and graph-based orderings. Evaluation results using 0.8 million geotagged Twitter users over a 3-month period demonstrate that GeoFind outperforms state-of-the-art approaches, with significant reduction of estimation error (25% of average error, 66% of median error). Given the improved location accuracy through the fusion of multiple data types in OSNs and possibly other data sources, new location privacy concerns arise and existing location privacy policies may need to be revisited and adjusted accordingly.

Chapter 5

Event Modeling

5.1 Introduction

With the fast growth of online population and rapid development of Web 2.0 technologies, online social media networks (OSMNs), which leverage both media and social networking by supporting easy web publishing and social interactions of online users, have become increasingly popular. A large amount of social media content is being generated by individual users on a daily basis. For instance, users of Twitter [69, 80], a popular microblogging social media site, send 140 million tweets per day. Moreover, OSMNs provide great opportunities for users to participate anytime and anywhere. Such user-based, real-time content generation is usually event driven. As events happen and evolve over time, users stay informed by seeking and sharing information through their social contacts (e.g., "following" and "follower" networks in Twitter). As a result, OSMNs have become the online gathering place for public engagement when real-time events happen and offer unique new opportunities for tracking and analyzing events. This has been demonstrated in various application domains, such as disease surveillance [19] and hazardous situations [65]. By sharing and receiving information among trusted and/or close social contacts, information related to specific events can be generated and disseminated in a highly effective and efficient fashion.

However, such user-generated event-related information in OSMNs is usually unstructured, and it is very difficult for individual users to capture a complete yet concise structural view of events using their social network-based information propagation channel. Moreover, as ongoing events evolve quickly and new messages are generated, the structural view of events should be adjusted to reflect the new developments in a real-time fashion. For instance, at Twitter, over 1 million tweets were generated by over 460,000 users in

128 days about the movie Avatar and every second, there may be some new updates about the event. As a result, users are constantly swamped by long streams of unstructured, redundant, and sometimes irrelevant messages, while at the same time lacking a comprehensive and well-organized view of events. **Event modeling**, which aims to identify inherent, evolving event structures and potential causal relationships, has become increasingly important for OSMNs and has the potential to significantly enhance our capabilities for information and knowledge management.

Event modeling for OSMNs is a challenging problem due to the following reasons:

- First, messages posted by users at social media sites tend to be short. For example, each tweet message has a maximal length of 140 characters. Also, messages generated by individual users tend to be unstructured, informal and differ in writing style. Such data sparseness, lack of context, and diversity of vocabulary make it difficult for traditional text analysis techniques to capture the semantic similarity among different messages [52].

- Second, different events may enjoy different popularity among users, and can differ significantly in content, number of messages and participants, time periods, inherent structure, and causal relationships [55].

- Third, large amounts of event-related information are continuously generated by OSMN users in real time. The event modeling process needs to be highly efficient, and incremental such that new information can be quickly incorporated into the event structure model.

To address these challenges, we have developed **ETree**, an effective and efficient event modeling solution targeting event-related information generated in OSMNs. Given all messages related to a specific event, ETree identifies the major **themes** (different aspects) of the event, the key message clusters (**information blocks**) and their hierarchical structure within each theme, as well as possible causal relationships (i.e., one led to the other) between information blocks. For example, people who are interested in the Haitian earthquake event may want to track various aspects of the event, such as new statistics, rescue efforts, donation information, etc. Our solution provides updated snapshot of the event in an easy-to-read hierarchical

tree structure, along with identified causalities within the event structure. Specifically, our work makes the following key contributions:

(1) An n-gram based content analysis technique for identifying core information blocks from a large number of short messages;

(2) An incremental and hierarchical modeling technique to efficiently identify and construct event theme structures at different granularities, which can be dynamically adjusted as events evolve;

(3) An improved event life cycle analysis technique for identifying potential causalities between information blocks;

(4) A detailed evaluation study using 3.5 million tweets over a 5-month period, which demonstrates the effectiveness and efficiency of the proposed solution.

5.2 Problem Formulation and System Overview

The problem of event modeling for OSMNs can be decomposed into several sub-tasks. Given a specific event, we first collect event-related information/messages via keyword-based search (more details of this process is described in Section 5.6). How to detect events is beyond the scope of this work. With scattered messages related to a certain event, we firstly cluster messages into **information blocks** (with high efficiency) to gain a basic understanding of the various "pieces" of an event (i.e., fundamental information units of semantically-similar messages). Next, to capture the overall structure of an event, we construct **hierarchical theme structures**, which represent the various aspects of an event at different levels of details. Using the identified information blocks as leaf nodes, we incrementally construct hierarchical theme structures for the event. Finally, we detect potential **causal relationship** between pairs of information blocks. For instance, rain drenched quake survivors in the tent camps may lead to people appealing to help those slum refugees. Identifying such causal relationships within an event structure helps us to better understand how an event evolves over time. Figure 5.1 illustrates the workflow and key tasks of event modeling for OSMNs. A formal definition of the event modeling problem is given as follows:

Event Modeling Problem: Let E be an event and

$$E = \{(m_1, t_1), \cdots, (m_i, t_i), (m_{i+1}, t_{i+1}), \cdots\},$$

where (m_i, t_i) refers to a message m_i posted at time t_i, and messages are sorted in ascending temporal order (i.e., $t_i < t_{i+1}$). Our goal is to organize these messages into an augmented, hierarchical event tree structure, i.e., $E \Rightarrow X = \{B, H, C\}$ that consists of the following:

- **Information blocks** B**:** Each event contains a number of information blocks $B = \{b_1, b_2, \cdots\}$ and each information block $b \in B$ contains multiple messages that are semantically coherent, i.e., representing a specific semantic meaning.
- **Hierarchical theme structures** H**:** $\{B_1, B_2, \cdots\}$ are combined hierarchically and at the highest level, an event can be represented by a set of theme structures $H = \{h_1, h_2, \cdots\}$, where $h_1, h_2, \cdots$ have zero or minimum similarity. Each theme structure $h \in H$ is a hierarchical subtree of X with a few information blocks as the leaf nodes. The whole hierarchical theme structures will be adjusted as the event evolves.
- **Causal relationships** C**:** For two information blocks $b_i, b_j \in B$, if b_j is caused by b_i, then $(b_i, b_j) \in C$.

5.3 Information Block Identification

Given a stream of messages that are related to a specific event, our first step is to group these messages into information blocks such that each block contains messages that share (almost) the same semantic meaning. Combining similar messages efficiently, as the goal of this step, will reduce the number of information units which will be used in the next two steps and is essential for real-time event modeling. Note that users of OSMNs can generate a huge number of messages (e.g., 140 million tweets per day at Twitter). Moreover, these messages are usually short, unstructured, and represent different writing styles of individual users. Simply clustering messages based on their cosine similarity is infeasible due to its low efficiency in

large-scale message processing. To address this issue, we propose a technique that considers **key phrases** in event-specific messages.

5.3.1 N-Gram Based Block Identification

As messages are propagated in OSMNs and new messages are generated, people tend to reuse the key phrases about an event. This is similar to the traditional "word of mouth" model, in which information is passed from one person to another. While the exact wording may be different, the key phrases remain the same. This is particularly true in online publishing, as users can easily copy-and-paste digital content. By identifying such key phrases in event-specific messages, we can then identify the core information blocks of an event.

Specifically, we propose an n-gram based content analysis technique, which works in two stages. The first stage detects frequent word sequences (i.e., n-grams, or key phrases) among a large number of event-related messages; each frequent sequence represents an initial information block. In the second stage, messages that are semantically coherent are merged into the corresponding information blocks. N-gram based techniques have been studied previously and are considered effective and efficient for identifying word patterns in documents [12]. Given the short length of the messages people generate at OSMNs, it is important that we choose the appropriate n, which is the minimum word sequence length. Similar to the work by Leskovec et al. on identifying key phrases from news articles [46], we choose $n = 4$ as it performs well in our experiments.

Once we have identified the frequent n-grams and their corresponding information blocks, we consider the remaining messages, i.e., messages that do not contain any of the frequent n-gram patterns. For each of these messages m_i, to measure the similarity between message m_i and an information block b_j, we consider the words that belong to both m_i and b_j and calculate their TF-IDF [63] weights in b_j. We then compute the weighted cosine similarity between each message and each information block. Messages that have high similarity with some information blocks are merged into the corresponding information block. In addition, messages that belong to a specific "conversation thread" (e.g., tweets "in reply to" other tweets) are merged into the corresponding information block.

It is important to note that the addition of messages into information blocks changes the list of words in each block and the TF-IDF weights of later messages. To address this issue, we design a staged and iterative merging process that considers messages in the order of descending similarity. In other words, we consider different ranges of similarity values. Starting from the highest similarity range, we merge messages in batches and update the similarity values of remaining messages at the end of each batch. This process helps to ensure that messages are merged into the most similar information blocks.

For merging raw data into basic information blocks, there are still two issues: one is how to define the similarity between raw data and information unit; and the other is as to more and more raw data are assigned to some information blocks, the similarity between raw data and information unit may change.

But in a document, there are both key information and noise words existing. So it's necessary to differentiate them when calculating similarity.

In this iterative process, we need to decide how much raw data merged in every pass and when to stop the whole process. A changing similarity threshold is used to control this process: Firstly, the similarity threshold won't go down to very low because low similarity may mean the raw data don't belong to any information blocks; Secondly, we hope getting enough highly confident raw data merged firstly to make sure the keywords' similarity become stable as soon as possible, so the high similarity threshold should decrease slowly in initial stages; At last, the highest similarity threshold depends on different events which at least make sure there are raw data assigned.

Based on these conditions, we define the changing similarity threshold as a arccotangent trigonometric curve (seen in Equation 5.1).

$$min + \frac{arccot\left[x * k + cot(max - min)\pi\right]}{\pi} \tag{5.1}$$

where min and max are two parameters stand for lowest and highest similarity threshold. and to let the granularity adjustable, we add another parameter k.

5.3.2 Conservation-Based Block Identification

The above n-gram based block identification focuses mainly on the textual content of event-related messages, and may miss out messages that are syntactically different yet semantically similar. In particular, in OSMNs, as users interact with each other and participate in event-related discussions, many messages are created **in reply to** existing messages. Recall that the messages provided as input to our system contain not only timestamp information, but also "conversation" information. Starting from an initial message, all other messages are (directly or indirectly) in reply to that message can be identified easily. This set of messages can be referred to as a "conversation thread". Such conversation-style relationships between messages provide valuable information with regard to the semantic similarity of messages.

Based on this observation, we propose to further enhance the n-gram based information blocks we have identified by considering messages with conversation information. We first identify messages that belong to the same thread of conversation. For each cluster of conversation messages, if most of the messages already belong to a certain information block identified by n-gram based algorithm, the remaining messages in this cluster are also merged into that block. Otherwise, this cluster of conversation-thread messages forms a new information block. While the n-gram based content analysis step leverages the syntactic text similarity, this conversation-based block identification step complements the n-gram based step with more semantic information, and improves the overall coverage and quality of the information blocks we can identify.

5.4 Incremental Hierarchical Theme Structure Construction

In this section, we present our design for constructing hierarchical theme structures using the information blocks we have identified. We describe first the static construction process, then the incremental process which integrates newly-generated messages into the hierarchical theme structure to keep the event structure up to date.

As we have mentioned, each event may consist of multiple themes representing the different aspects of the event, such as "rescue" and "donation" of the Haitian earthquake event; "cast", "animation" and "reviews" of the movie Avatar. Generally it is difficult to discover themes agreed by everyone, since themes

could be defined differently by different people. So again a simple clustering approach does not meet the needs. Instead, each theme can be represented as a tree structure with information blocks as the leaf nodes and subtopics as the internal nodes. Such hierarchical theme structures enable a systematic organization of event-related information that is comprehensive yet concise, and allow users to explore an event from different aspects and at different granularities.

Algorithm 4 $HierarchicalStructure(B)$

Input: set of information blocks B
Output: hierarchical theme structures H
$H = \phi$
for each block $b_j \in B$ **do**
  create node $n_j = \langle b_j, M_j \rangle$; add n_j to H
end for
$< n_i, n_j, s_{i,j} >= maxSim(H)$
while similarity $s_{i,j} > 0$ **do**
  create a new parent node n_p for n_i and n_j
  $ReStructure(n_p)$; add n_p to H
  $< n_i, n_j, s_{i,j} >= maxSim(H)$
end while
add a virtual $root$ node; return H

5.4.1 Static Theme Structure Construction

In a hierarchical theme structure, child nodes contain more specific information while parent nodes are more general and may represent the common topic of its child nodes. For instance, a parent node about "donation for earthquake" may have child nodes talking about "U.S. donation for earthquake" and "China donation for earthquake". Intuitively, the desired hierarchical theme structures should satisfy the following properties:

(1) A parent node's meaning should be more general than that of its child nodes and the difference should be significant enough;

(2) Nodes with similar meanings should be sibling nodes;

(3) Meanings of sibling nodes should not be the same or the subset of one another.

Algorithm 5 $ReStructure(n_p)$

```
Input: a new parent node n_p
for each internal child node n_i of n_p do
  if M_i == M_p then
    remove n_i and attach all its children to n_p
  end if
end for
for each child pair ⟨n_i, n_j⟩ of n_p do
  if M_i ⊃ M_j then
    attach node n_i to n_j as its child
  else if M_i == M_j then
    if n_i and n_j are both internal nodes then
      attach n_i's children to n_j; remove n_i
    else if n_i is leaf node ∧ n_j is internal node then
      attach n_i to n_j as its child
    end if
  end if
  ReStructure(n_j) if n_j has new child
end for
```

Definition 1. *A leaf node's* $Meaning$ M_i *is the set of keywords* K_i *of its corresponding information block* b_i*; and an internal node's* $Meaning$ M_i *is the intersection of its child nodes' meanings.*

The set of keywords for each information block can be obtained by selecting the words with high TF-IDF weights. Two nodes are considered different if their $Meaning$ contain different sets of keywords.

Algorithm 4 shows the process of constructing the hierarchical theme structures. Starting with the information blocks as the leaf nodes, this process iteratively selects two nodes ($i \neq j$ and nodes n_i, n_j have no parent node) with the highest similarity using the $maxSim(H)$ procedure, and merges the two nodes into a new parent node, thus ensuring Property 2. This new parent node is then restructured (Algorithm 5) to ensure Property 1 and 3. First, if an internal child node has the same meaning as the parent node, that child node is removed and its children are attached to the parent node. Next, we examine sibling nodes. If one node's meaning is more specific than that of its sibling node, that node becomes the child of its sibling. If two sibling nodes have exactly the same meaning, then the two sibling nodes and their children are merged into one. This restructuring process continues until all nodes in the hierarchical theme structures satisfy all three properties discussed above.

The most time-consuming operation of this algorithm is the recursive restructuring process. Nevertheless, it is a local update process (only the subtree of n_p need to be restructured), which makes the algorithm more efficient than techniques that require global updates, where the whole tree need to be restructured every time a new node is added [74].

5.4.2 Incremental Structure Construction

As events happen and evolve over time, a large amount of event-related information is continuously generated by the users of OSMNs. To keep the event models up to date, newly-generated messages need to be integrated into the models in a timely fashion. It would be time-consuming and extremely wasteful if we have to reconstruct the whole structure from scratch each time a new message is added. To address this issue, we propose an incremental modeling process to maintain dynamic hierarchical theme structures.

Newly-generated messages about an event either focus on some existing topics, or contain new topics about the event. In the former case, these new messages can be easily merged into existing information blocks. While in the latter case, new information blocks need to be created; we then need to determine where to (in the hierarchical theme structure) to place the new blocks and adjust the overall hierarchical theme structures as needed.

To handle these changes, our incremental structure construction algorithm is designed to utilize a top-down update process: (Algorithm 6): Given a new message m, we first check n_p (initially, n_p is the $root$ node), and its child nodes to select the one that is most similar to m. Note that the $Similarity()$ function is the same as the weighted cosine similarity we define in Section 5.3. If the most similar child node is an internal node, this process continues recursively until the node most similar to m is either n_p or a leaf node. If the similarity value is higher than a threshold δ, message m is merged into that node. Otherwise, a new node is created that contains only the new message, and the new node is added as a child (or grandchild) of n_p. After m is inserted, the $ReStructure()$ procedure is called on n_p to restructure its subtree and ensure that the three properties are still maintained.

5.5 Causal Relationship Detection

Given the information blocks we have identified and the hierarchical theme structures we have constructed, one more question we want to answer in the event modeling process is whether there exists any causal relationships between information blocks. Understanding such causal relationships is important as it provides insights into how an event evolves through multiple stages and how these stages impact each other. However, finding exact causal relationships is a very difficult task without incorporating domain knowledge [55]. Previous research [55, 13] has shown that two pieces of information are more likely to be causally related if they are similar in content. Also, the study by Yang et al. [77] shows that the relevancy of two pieces of information increases when they are temporally closer to each other. Based on these observations, we aim to tackle our problem of causality detection in OSMNs by considering **both content similarity and temporal relevance**.

Given two information blocks b_i and b_j, let S and T be the content similarity and temporal relevance between these two blocks, respectively, we define the causal relationship C as a function of S and T, i.e.,

$$C = S \times T \tag{5.2}$$

For content similarity S, we use the same weighted cosine similarity as defined in Section 5.3. Next, we focus on defining the temporal relevance T.

Using the timestamped messages belonging to each information block, we can calculate the following temporal information of the block: (1) **start/end time**, which correspond to the timestamps of the earliest and latest messages in the block; and (2) **temporal life cycle**, which is a temporal distribution reflecting the number (or percentage) of messages posted within each time period. As the examples shown in Figure 5.2, by examining the temporal intersection of two information blocks, we can determine their temporal relevance in 2 stages. Considering two blocks, one on the left and one on the right of the time line, are approaching towards each other from far away. Let **critical point** be the point when two blocks overlap with each other and have the maximal temporal relevance, then

- Stage 1: Before the intersection reaches the critical point, the temporal relevance should gradually increase from minimum to maximum. In addition, the increasing speed should be different when

there is no overlap (called stage $1(a)$) and when the intersection is smaller than the critical point but greater than 0 (called stage $1(b)$);

- Stage 2: After passing the critical point, the temporal relevance should gradually decrease. And when they become completely parallel (i.e., happening at the same time), the temporal relevance (and the causal relationship) decreases to minimum.

Then the question is how to determine the changing speed in different stages. Based on the analysis by Chen et al. [13], we assume that stage $1(b)$ and stage 2 have the same linear changing speed as to the intersection but the former is positive and the latter is negative. And the changing speed in stage $1(a)$ follows the inverse proportion curve.

Based on this intuition, we define the *temporal relevance* T between block b_i and b_j in Equation 5.3, where b_i^s and b_i^e are the start and end time of block b_i, f_t is defined as the intersection frequency of two blocks at time point t, and the range of the temporal relevance is (0,1]. Parameter $2 * \theta$ $(0 < \theta < 1)$ defines the value of the critical point. In our experiments, θ is set to 0.2 based on the power law property analysis of message generation frequency in the life cycle of information blocks.

$$T = \begin{cases} 1/\left(b_j^s - b_i^e + \frac{\theta-1}{2\theta-1}\right), & b_j^s > b_i^e \\ \frac{1}{\theta-1} * (\frac{F}{2} - 1), & 2\theta \leq F < 2 \\ \frac{1}{\theta-1} * (2\theta - 1 - \frac{F}{2}), & 0 < F < 2\theta \end{cases} \tag{5.3}$$

where intersection

$$F = \sum_{t=b_j^s}^{min(b_i^e, b_j^e)} f_t \tag{5.4}$$

One work that is closely related to our causality relationship detection design is the $TSCAN$ method proposed by Chen et al. [13]. Here, we highlight the difference between our method and theirs. Firstly, the causal relationships defined in $TSCAN$ are based on blocks in two different themes and blocks within one theme are connected as a causal time line. Usually, blocks in different themes can be very different in

Table 5.1: List of Events Used in Evaluation

ID	Event	Category	#Tweets	#Users	#Days
1	Possibility and impact of China allowing its currency rising	business	718	355	14
2	MySpace's CEO was fired for not very good achievement	business	1,749	1,022	11
3	Processed foods were recalled for containing HVP	health	2,526	1,398	26
4	Apple filed a lawsuit against HTC for patent infringement	business	4,908	2,576	18
5	Duke defeating North Carolina in a match of NCAA	sports	6,279	5,321	6
6	President Obama proposed a bank reform to save financial system	politics	8,061	4,965	76
7	Many airports used body scanning to detect bombs	world	9,088	5,863	57
8	A global outbreak of H1N1 influenza virus from 2009	health	12,138	3,832	46
9	An TV series "Modern Family" were renewed for a second season	entertainment	18,692	10,549	33
10	Will Google leave China?	business	24,024	9,571	70
11	A magnitude 8.8 earthquake stroke Chile on February 27, 2010	world	31,814	13,712	41
12	Toyota had to recall millions of vehicles globally	business	43,626	15,345	71
13	Obama health care reform was passed to become law	politics	54,074	19,134	38
14	Google launched the extended release of Google Wave	science/technology	58,511	30,505	114
15	The 82nd Academy Awards ceremony was held on March 7, 2010	entertainment	87,449	49,356	33
16	2010 NBA all star game	sports	112,318	57,174	27
17	Google announced a social networking tool Google Buzz	science/technology	297,972	117,146	103
18	Apple released a table computer called iPad	science/technology	410,297	112,300	101
19	A magnitude 7.0 earthquake stroke Haiti on January 12, 2010	world	479,982	180,076	99
20	A highly popular epic science fiction film in 2009: Avatar	entertainment	1,001,530	463,128	128

content and most of the causal relationships identified across themes may not be useful. While within the same theme, potential actual causality between concurrent blocks is ignored by $TSCAN$. Secondly, when defining the time relevance, $TSCAN$ only utilizes the start and end time information of blocks, while in our design, we leverage the detailed temporal life cycle information of blocks. Our design can achieve better characterization of temporal relevance and causal relationships than $TSCAN$, as demonstrated in the two cases in Figure 5.2.

In the left part of Figure 5.2, two blocks appear in chronological sequence which tells us these two blocks probably have strong temporal relevance. However, according to the definition in $TSCAN$, these two blocks are very close to each other when only considering their start and end time, so they are thought to be parallel and their temporal relevance is considered weak. But our design can correctly capture the strong temporal relevance by considering the small intersection of the two blocks' life cycles. In the right part of Figure 5.2, according to the definition in $TSCAN$, the temporal relevance is strong because the bottom block seems to appear directly after the top one. Actually their climax almost overlap, which means they probably happened in parallel, and therefore unlikely to contain causal relationship.

5.6 Experimental Evaluation

In this section, we evaluate ETree, the proposed event modeling solution for OSMNs, using real-world events and event-related messages generated by individual users. Our evaluation aims to answer the following questions:

- Does our n-gram based information block identification algorithm generate coherent information blocks with good content coverage of each event?
- Does the hierarchical theme structure capture the various aspects of an event at the appropriate granularities?
- Does the incremental modeling process achieve high efficiency and generate good-quality intermediate results?

- Does our causal relationship detection algorithm achieve high accuracy with regard to the identified causalities?

5.6.1 Dataset Description

The data used in our experiments are real-world messages gathered from Twitter, one of the most popular online social media networks. Using Twitter's APIs, we have collected event-related information over a 5-month period. To ensure diversity and scalability of the evaluation data set, we manually selected 20 events spanning 7 different categories, including World, Politics, Business, Health, Entertainment, Science/Technology and Sports. The messages related to each event are collected using the keyword-based text search API provided by Twitter. Specifically, for each event, we handpick a set of keywords and use the Twitter API to collect all tweets that match at least one of the keywords. We also collected tweets that belong to the same conversation threads as the tweets returned by the search API. A summary of the 20 events is listed in Table 5.1. In total, our data set is around 75GB in size and consists of 3.5 million tweets.

5.6.2 Quality of Information Blocks and Theme Structures

First, we evaluate whether our n-gram based information block identification algorithm can capture the main content of an event, and whether the identified hierarchical theme structures have good quality.

We use $Coverage$ as the metric to evaluate the effectiveness of the information block identification algorithm. $Coverage$ of an event is defined as the percentage of messages which are captured into one of the identified information blocks. To calculate $Coverage$ of an event, we calculate the sum of the number of messages in all the information blocks of that specific event, and divide it by the total number of messages retrieved for that event. For the 20 events used in the evaluation, the information blocks identified by our method has high $Coverage$, ranging from 71% - 92% (84.2% on average).

Besides information block $Coverage$ of an event, another measure of the quality of the event structure is the relevance of the information blocks. Intuitively each theme in an event should represent a certain aspect. To evaluate the quality of identified themes of an event, we define $Coherence$ as the percentage of coherent themes in an event and a theme is considered coherent if more than half of its information blocks

are relevant. To calculate $Coherence$ of an event, we manually examine the relevance of information blocks in each theme. ETree has identified highly coherent themes for almost all the 20 events, with $Coherence$ values ranging from 63% to 82% (76.9% on average).

5.6.3 Efficiency of Incremental Event Modeling

Next, we evaluate whether the incremental modeling process of ETree achieves high efficiency and generates high-quality intermediate theme structures at the same time.

We compare the computation time of generating hierarchical theme structures, using three different algorithms: ETree, ETree without incremental modeling (ETree-NI), and TSCAN [13]. TSCAN is a popular algorithm widely used, which derives an event's major themes from the eigenvectors of a temporal block association matrix. Because neither ETree-NI nor TSCAN supports incremental modeling, each increment of the tweets would cause the system to re-compute the entire hierarchical structure. While in ETree, we only update the original structure by incorporating the newly-created data. Since the non-incremental algorithms take a long time to run for each event, we choose three events with different size, content and structure for this evaluation, including Event 7, 12 and 19.

Figure 5.3, 5.4, 5.5 shows the computation time in seconds for the three event modeling algorithms. The results suggest that the computation time of ETree is stable and much lower because the main influential factor is the number of newly-created tweets, while the computation time increases very quickly for ETree-NI and TSCAN. For example, for event 12, the computation time of ETree is between 100 seconds to 800 seconds for any 10% of the tweets, while the computation time of ETree-NI increases from 300 seconds to 12600 seconds and TSCAN from 2,000 seconds to 83,700 seconds as the number of tweets increases. Apparently, this dramatic increase makes it difficult for ETree-NI or TSCAN to generate up-to-date theme structure in short time intervals when the number of messages about an event becomes large. Note that we are evaluating the efficiency of the algorithms on a single core machine with limited memory, which means the absolute value of excecution time will be improved significantly within state-of-art hardware environment.

5.6.4 Quality of Detected Causal Relationships

We also evaluate how well our causal relationship detection algorithm works for different events. Since it is difficult to manually label all actual causal relationship pairs as the ground truth, we use TSCAN and ETree to compute causal relationship pairs first, then manually verify these pairs. To reduce the influence of subjective factors in the verification process, two researchers worked independently to cross check the results. When we consider the quality of the identified causal relationship pairs, events with a small number of identified causalities are prone to random noises (due to the limited size and information) and may easily skew the overall results. Instead, we only consider 11 popular events (event ID from 10 to 20) and report their $F1 - measure$ values in Figure 5.6. We can see that ETree outperforms TSCAN by 49% on average for all these events.

5.6.5 Case Study

Next, we use the "Haitian earthquake" event as an example to demonstrate the quality of our theme hierarchy in more detail.

5.6.5.1 Hierarchical Theme Structures

We choose several easy-to-understand themes in this event and show the structures in Figure 5.7. Two observations can be drawn from the theme structures: (i) most information blocks in a theme have relevant content; and (ii) the hierarchical theme structure clearly reflects the level of granularity of the theme. For example, the theme chosen in the event "Haitian earthquake" talks about the rainy season after earthquake. It contains four pieces of detailed information: scientists' prediction, rain adding to misery, camps needed and only one piece plastic for every Haitian family. Each piece of the information is followed by more detailed messages. For example, block 6, 7 and 8 are more specific discussions about the content of "rain adding misery".

5.6.5.2 Causal Relationships

Next, we examine the causal relationship pairs that ETree has detected in this event. From the themes shown in Figure 5.7, we can easily find some causality pairs by reading the content. For example, block 7 talks about survivors in Haiti suffering from the rain and block 6 talks about people appealing for help. Block 7 began on Feb 12th and ended on March 14, while block 6 was from Feb 15th to March 20th. When only considering the start and end time, these blocks seem to have happened in parallel. However, when considering the life cycle distribution, we can see that the climax of block 7 was from Feb 12th to March 1st and the climax of block 6 was from March 14th to March 18th. This means when block 7's popularity began to decrease, block 6 began to become popular. These two blocks actually did not occur in parallel. This causal relationship pair clearly demonstrates the improved accuracy of ETree, compared with prior mechanisms such as TSCAN (see the first case in Figure 5.2, Section 5.5).

5.7 Conclusions

This work presents **ETree**, an effective and efficient event modeling solution for real-time and information-intensive online social media networks. ETree utilizes an n-gram based content analysis technique to group a large number of event-related messages into semantically-coherent information blocks, an incremental modeling process to construct hierarchical theme structures, and a life cycle-based temporal analysis technique to identify potential causal relationships between information blocks. Detailed evaluation results using 20 real-world events and 3.5 millions tweets demonstrate that ETree can generate high-quality event structures with high efficiency. We anticipate to apply ETree to larger and more noisy dataset and identify new research problems. We are also interested in exploring more usage of social ties to improve the quality of ETree.

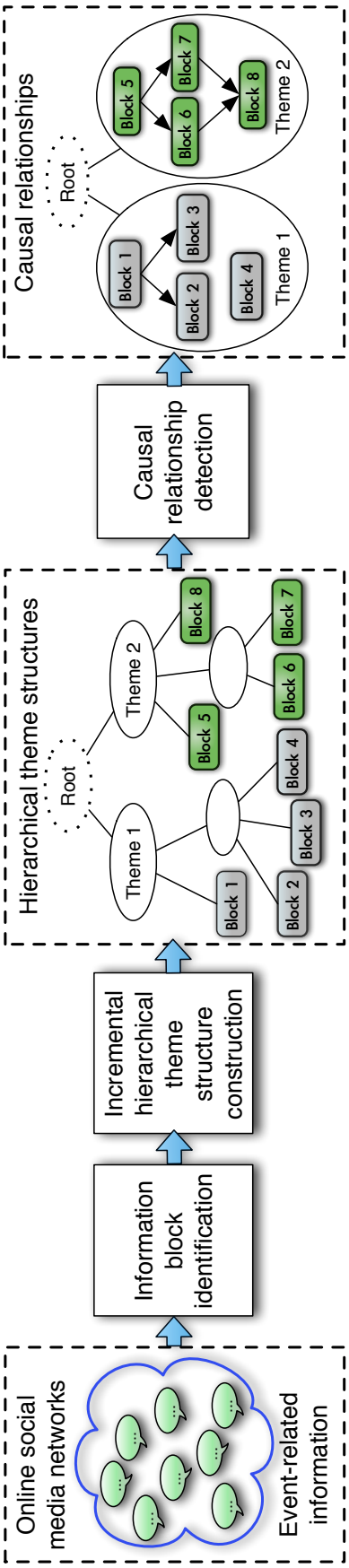

Figure 5.1: ETRee: Effective and efficient event modeling for real-time online social media networks — System overview.

Algorithm 6 $IncrementalStructure(n_p, m)$

Input: a node n_p in event structure H; a new message m
if n_p == $root$ **then**
  for each leaf node n_i in H **do**
    if m belongs to a conversation in n_i **then**
      merge m into n_i and algorithm ends
    end if
  end for
end if
$xSim = Similarity(m, n_p)$; $xNode = n_p$
for each child node n_i of n_p **do**
  if $Similarity(m, n_i) > xSim$ **then**
    $xSim = Similarity(m, n_i)$; $xNode = n_i$
  end if
end for
if $xNode$ is an internal child node **then**
  $IncrementalStructure(xNode, m)$
else
  if $xSim >$ minimum similarity threshold δ **then**
    merge m into $xNode$
  else
    create a new leaf node n_j, only containing m
    if $xNode$ == n_p **then**
      attach n_j to n_p as its child
    else
      create a new internal child n_k of n_p
      attach $xNode$ and n_j to be n_k's children
    end if
    $ReStructure(n_p)$
  end if
end if

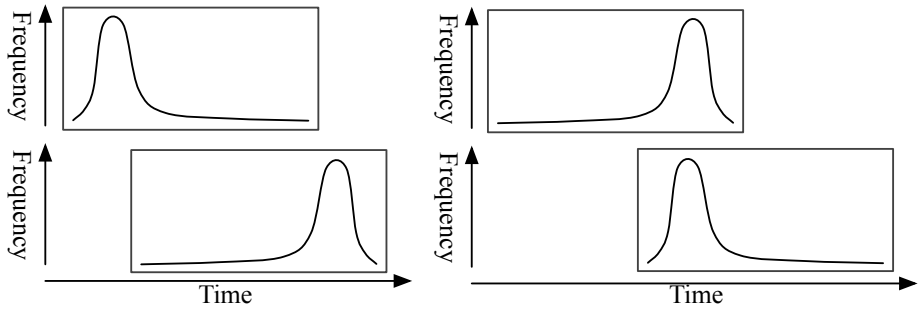

Figure 5.2: Two cases demonstrating when ETree outperforms TSCAN.

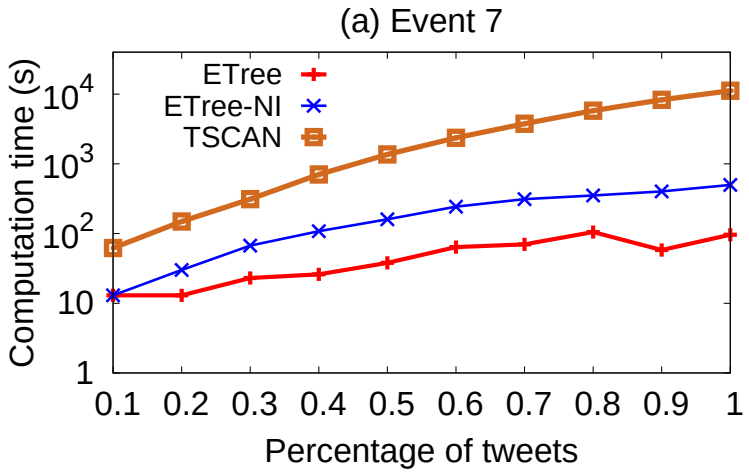

Figure 5.3: Event 7: Comparison of computation time of ETree, ETree-NI and TSCAN.

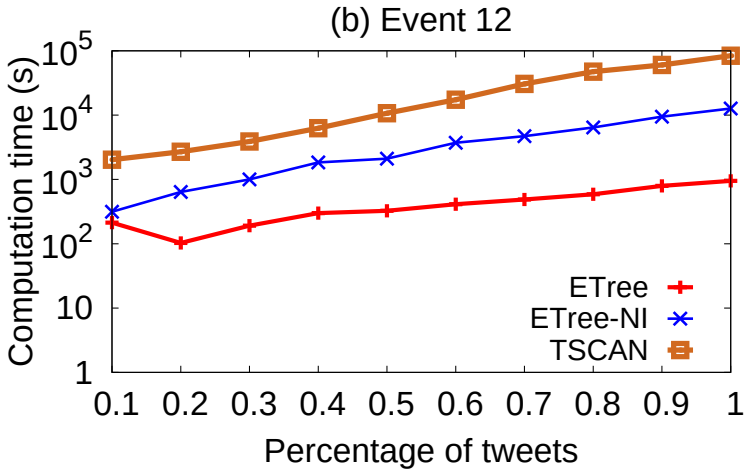

Figure 5.4: Event 12: Comparison of computation time of ETree, ETree-NI and TSCAN.

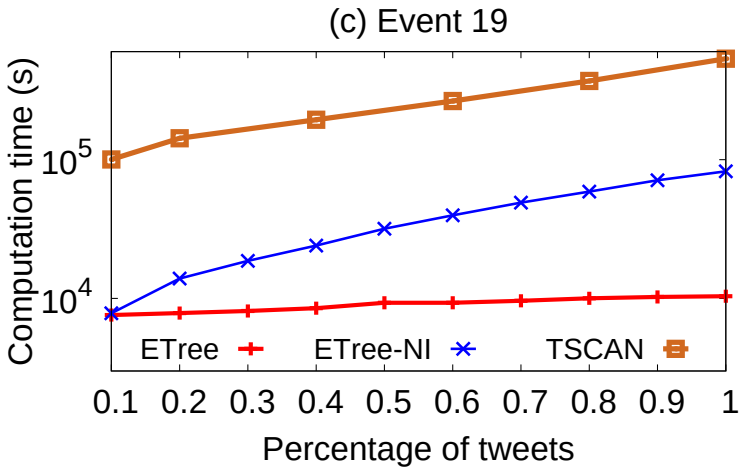

Figure 5.5: Event 19: Comparison of computation time of ETree, ETree-NI and TSCAN.

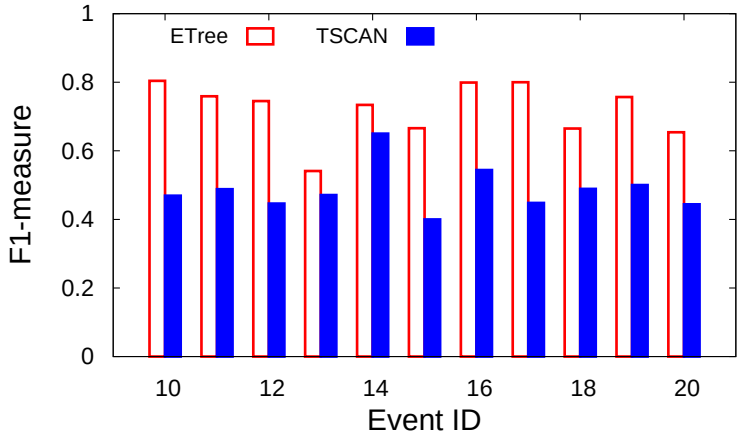

Figure 5.6: Causal relationship pairs detected by ETree and TSCAN

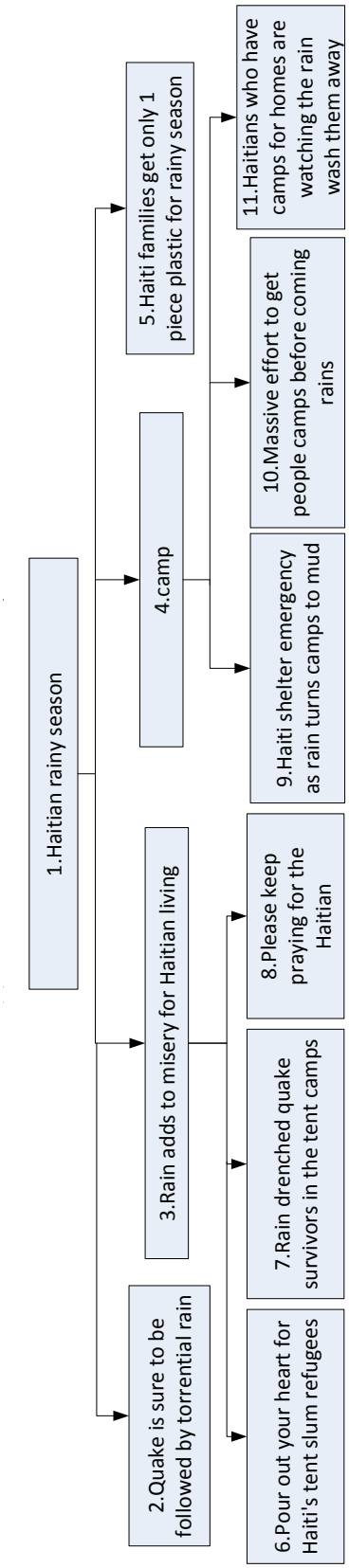

Figure 5.7: Partial hierarchical theme structures constructed by ETree for the event “Haitian earthquake”.

Chapter 6

Applications and Usage Scenarios

Event analysis in online social networks has broad applications and usage scenarios. In the following sections, we demonstrate them by going through all three research problems as we explained above. At the end, we show some full usage scenarios with the combination of all the problems for proof-of-concept purpose.

6.1 Event Context Identification

As described in section 3, we can successfully characterize a group of users who participate a certain event. Event context has potentially the following applications and usage scenarios.

6.1.1 Event Correlation

There are enormous discussions on OSNs when presidential related events happen. From section 3, in our evaluation dataset, we find *2013 Obama inauguration* is highly correlated with event *2013 The International Consumer Electronics Show (CES)* after identifying event context. This interesting finding reveals the fact that the group of people who discusses the Obama inauguration event is similar to the group of people who are interested in technology and discuss the CES event often. Usage scenarios include better understanding of voters and election policy improvement.

There are also many brands or organizations related events. When companies keep track of events related to them and figure out their contextual correlation, potential value can be produced. For instance, we imagine that a new generation of tablets were released and being sold well on Amazon.com covered

by some media. At the same time, lots of people were discussing the massive amount of tablet pictures from Pinterest.com. Events are monitored by two companies on their sides and by applying the event context identification techniques, they can discover the correlation between these two events and potentially increase the possibility of business collaboration.

Celebrities may constantly follow events happening on OSNs. They are especially interested in attracting fans with public exposure in the right direction. Imagine that people are discussing several top entertainment shows they like watching on OSNs and it would be very useful for celebrities to know which show to attend to maximize the influence on their fans. With the help of identified event context, the group of users who are talking about certain show can be easily identified, and one can find the right entertainment show to attend by matching fans' interests.

6.1.2 User Recommendation

During crisis situations, there is a common need to find relevant people and keep them updated. Based on a set of identified event contexts from a collection of events, we can quickly extract the context of a particular on-going crisis event. Then relevant users can be recommended to receive updates of the event. For instance, during *Haitian Earthquake* event, family members or friends of the people live in Haiti would be kept updated if event context identification techniques are applied. One way to evaluate the accuracy of the recommendation is the prediction accuracy of users. As shown in section 3.4.2, we achieve 90% of the best prediction performance with only 10% of the training users.

6.1.3 Friendship Recommendation

Friendship recommendation is a popular task in OSNs. Either the platform itself, or third-party applications want to intelligently find users potential friends to recommend. Existing approaches focus on understanding either social structure of users or based on some detected communities. A new idea is to leverage users' participation of event discussion on OSNs and recommend new friends to a user if they participated in similar events. In other words, in order to recommend friendship, one can identify the event context of a set of events and identify users who are related to similar event context. As shown in 3.4.5,

users are more likely to be friends if they share similar event context.

6.2 Location Inference

The location inference technique helps to identify users' location when events happen. With city-level inference accuracy, we can apply the technique into different scenarios.

6.2.1 Event Discussion Location

Brands or companies follow events mentioning their name on OSNs for analytics purposes. In most cases, they want to know where they are being discussed. There are a lot of benefits to identify event participants' location with the help of location inference techniques. For marketing purpose, knowing event participants' location can help company build better advertising policies, which will further affect sales. For instance, if a company discovers their product being discussed in certain areas before the release of the product, it may make sense to find a local influential person to make advertisement. The company sales people may tune their sale policy to meet the requirement of that area. Furthermore, location can be used for supply chain optimization and customer service policy change. Company may have better idea of where to store their products and where they should set up their customer service center.

6.2.2 Event Reaction

During crisis situations, it is very important to know where the event happens. With the help of location inference on OSNs, organizations can quickly find out the location of the crisis and take further actions. For instance, people may feel the earthquake earlier than news report in some areas like Japan, and publish the information on OSNs. Accurately identifying the location of the earthquake is very useful to send alert to people coming to this area, and deploy resources for the situation.

6.3 Event Modeling

Event modeling techniques help organize the hierarchical structure and causal relationships of the events. As a result, one can understand different aspects of events and how events evolve. This is particular

useful for scenarios where we need to dig into the event.

6.3.1 Event Story Analysis

Brands always want to keep track of events which mention their name. One important task is to follow the story of the events and have a good understanding of the structure. With the help of event modeling techniques, companies can conveniently decompose the events they are interested in into a hierarchy, from the most important themes to every detail. From the hierarchy, it is very intuitive to understand the whole event with the organized structure. Also, it makes the opinion analysis task easier by following the hierarchy of the event. By understanding the related events better, companies can potentially improve their product quality with consideration of customers' feedback.

6.3.2 Event Tracking and Prediction

Event prediction is one of the most challenging tasks in event analysis domain. By tracking historical events, it may be possible to extract some patterns when new similar events happen and predict the next step. This require to understand how historical events evolve and the internal causal relationships of the events. Event modeling techniques enable all the required analysis. For instance, in the *Haitian Earthquake* event, since from historical events we had learned that some tragedy was caused by heavy rain after earthquake, we can predict the next step was to deploy water-resist resources for the crisis situations.

6.4 Usage Scenarios

From the above description, we see many applications of our proposed event analysis techniques in OSNs. In this section, we will focus on two concrete usage scenarios which utilize all the techniques we propose in the thesis.

6.4.1 Company Brand Construction

With the rapid growth of OSNs, more and more companies are focusing on OSNs for their brand construction. Leveraging event analysis techniques, one can construct their brand more effectively.

Firstly, it is possible to build better marketing strategies. Event context identification techniques will help to extract the group of people who are interested in their brand. Also, their locations can be inferred via location inference techniques. With good understanding of whom and where the potential customers are, companies can make accurate advertising and potentially help to boost the popularity of their brand.

Secondly, companies can utilize event analysis techniques to help improve their product. By apply event modeling, different aspect of the event will be display and companies can conveniently conduct opinion analysis of the feedback from their customer on OSNs. Companies will have clear understanding of different opinion about their products and potentially improve the quality. As a result, their brands may become more popular.

Thirdly, better sales and customer service can be provided. Knowing the location of people who participate the brand discussion help to make decisions of where to put the customer service center. Also it helps to optimize supply chain for effective sales. Overall better brand will be constructed based on above factors.

6.4.2 Disaster Response

Recent research show that when crisis happens, OSN platforms are effective for event updates, public discussions and engagement. In particular, with event analysis techniques, better disaster response can be provided by monitoring the ongoing event. There are two cases as explained below.

Accurate updates delivery can be provided. By applying event context identification and location inference, it is possible to identify who to contact when crisis happens and where those people are. This is very useful when the disaster is not publicly available in news or other traditional media. Then local updates can be rapidly spreaded on OSN platforms based on identifies people and location information.

Better action taking and resource deployment policy can be made. With location inference, the location information of a event can be quickly identified and can be publicly available very quickly due to the real-time characteristics of OSNs. As a result, actions can be taken rapidly and resource can be deployed in time. By using event modeling, historical crisis situations can be analyzed and the evolution of the situations can be replayed. When new disaster happens, historical knowledge may be useful for better prediction of

next events and actions can be taken even before crisis situations happen.

Chapter 7

Conclusion

In this chapter, we summarize major contribution and components of the thesis and explore future directions along the line of this research.

7.1 Summary

With the growing popularity of Online Social Networks, massive amount of user-generated content is showing the willingness of event-related discussions. However, traditional social network infrustructures better support social interation better than presenting the disscussion in an event-centric view. This is a unqiue opportunity for creating event analysis techniques to support the transformation.

In this thesis, we propose three event analysis techniques in online social networks which aim to fill the gap between social interaction perspective to event-centric perspective. It consists of three major components:

- Event context identification. Given needed events, we can extract event context i.e. the groups of people who are interested in the events. We have presented a matrix factorization algorithm to solve the event context identification problem. It selects anchors from users' followers and incorporates anchor information into an extended PMF framework. Our evaluation using real-world Twitter data shows that our proposed technique outperforms existing matrix factorization techniques by 20.0%. We also show potential applications including users' participation in events, retrieval of relevant events, and friend recommendation based on identified event context.

- Location identification. Location identification aims to solve the problem of lacking geo-tagged content in OSNs. It geolocates OSN users by leveraging both textual and social structual information in OSNs. It then fuses results from the two sources to give a final, boosted result. Detailed evaluation results based on 0.8 million Twitter users and 21 million Tweets over a three-month period shows our approach outperforms previous methods.

- Event modeling. Event modeling aims to provide a structured view of user generated content related to a given event. It identifies information blocks, creates an incremental and hierarchical structure and discovers internal causal relationships. Detailed evaluation results using 20 real-world events and 3.5 millions tweets demonstrate that ETree can generate high-quality event structures with high efficiency.

7.2 Future Directions

We have proposed key techniques when transforming from social interaction perspective to event-centric perspective. In order to make the techniques more general to use in real world systems, we have several future directions to explore.

7.2.1 Richer Feature Set

All current approaches are based on most representative existing features. As the development of OSNs, it is possible that other features keep growing and worth adapting. For the event context identification problem, we would like to explore other potential features, such as location information in users' profiles and tag information from users' followers, and consider how these features can be used in our model for better event context identification. In the location inference framework, it is possible to explore location information in users' profiles, as users are putting more and more correct location in their profiles. Conversation information are more retrievalable recently. So during the event modeling process, we would leverage conversation context to generate more consistent event structures.

7.2.2 Larger Scale

As the number of OSNs is growing, we would like to explore new challenges when our proposed techniques use in larger scale. This require the algorithms to be highly robust, reliable and efficient. It would be interesting to deploy event context identification techniques to a cloud environment to run the algorithm in paralell. We anticipate to apply location inference to larger scale and explore the possibility of utilizing real-time properties of online social media networks. We would like to apply our event modeling solution to more events and expect more noise in the dataset and make sure the algorithm is robust and reliable enough. Larger scale will potentially help us identify new research problems.

Bibliography

[1] E. Agapie, G. Chen, D. Houston, E. Howard, J. Kim, M. Y. Mun, A. Mondschein, S. Reddy, R. Rosario, J. Ryder, A. Steiner, J. Burke, E. Estrin, M. Hansen, and M. Rahimi. Seeing our signals: combining location traces and web-based models for personal discovery. In HotMobile '08, pages 6–10.

[2] James Allan, Rahul Gupta, and Vikas Khandelwal. Temporal summaries of new topics. In SIGIR '01, pages 10–18, 2001.

[3] James Allan, Ron Papka, and Victor Lavrenko. On-line new event detection and tracking. In SIGIR '98, pages 37–45.

[4] Denise Anthony, Tristan Henderson, and David Kotz. Privacy in location-aware computing environments. IEEE Pervasive Computing, 6(4):64–72, October 2007.

[5] Lars Backstrom, Jon Kleinberg, Ravi Kumar, and Jasmine Novak. Spatial variation in search engine queries. In WWW '08, pages 357–366.

[6] Lars Backstrom, Eric Sun, and Cameron Marlow. Find me if you can: improving geographical prediction with social and spatial proximity. In WWW '10, pages 61–70.

[7] Louise Barkhuus, Barry Brown, Marek Bell, Scott Sherwood, Malcolm Hall, and Matthew Chalmers. From awareness to repartee: sharing location within social groups. In CHI '08, pages 497–506.

[8] Norbert Blenn, Christian Doerr, Nasireddin Shadravan, and Piet Van Mieghem. How much do your friends know about you?: reconstructing private information from the friendship graph. In SNS '12, pages 2:1–2:6.

[9] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In ICML '05, pages 89–96.

[10] Davide Buscaldi and Paolo Rosso. A comparison of methods for the automatic identification of locations in wikipedia. In GIR '07, pages 89–92.

[11] Giuseppe Carenini, Raymond T. Ng, and Xiaodong Zhou. Summarizing email conversations with clue words. In WWW '07, pages 91–100, 2007.

[12] William Cavnar and John M. Trenkle. N-gram-based text categorization. In SDAIR '94: Proc. of the 3rd Annual Symposium on Document Analysis and Information Retrieval, pages 161–175, 1994.

[13] Chien Chin Chen and Meng Chang Chen. TSCAN: a novel method for topic summarization and content anatomy. In SIGIR '08, pages 579–586.

[14] Ling Chen and Abhishek Roy. Event detection from flickr data through wavelet-based spatial analysis. In CIKM '09, pages 523–532.

[15] Zhiyuan Cheng, James Caverlee, and Kyumin Lee. You are where you tweet: a content-based approach to geo-locating twitter users. In CIKM '10, pages 759–768.

[16] A. Clauset, C.R. Shalizi, and M.E.J. Newman. Power-law distributions in empirical data. SIAM Review, 51(4):661–703, 2009.

[17] Brendan O Connor, Michel Krieger, and David Ahn. Tweetmotif : Exploratory search and topic summarization for twitter. In Artificial Intelligence, pages 384–385, 2010.

[18] Sunny Consolvo, Ian E. Smith, Tara Matthews, Anthony LaMarca, Jason Tabert, and Pauline Powledge. Location disclosure to social relations: why, when, & what people want to share. In CHI '05, pages 81–90.

[19] Courtney D. Corley, Diane J. Cook, Armin R. Mikler, and Karan P. Singh. Text and structural data mining of influenza mentions in web and social media. International Journal of Environmental Research and Public Health, 7(2):596–615, 2010.

[20] David J. Crandall, Lars Backstrom, Daniel Huttenlocher, and Jon Kleinberg. Mapping the world's photos. In WWW '09, pages 761–770.

[21] Justin Cranshaw, Eran Toch, Jason Hong, Aniket Kittur, and Norman Sadeh. Bridging the gap between physical location and online social networks. In Ubicomp '10, pages 119–128.

[22] Ravi Kumar Prabhakar Raghavan David Liben-Nowell, Jasmine Novak and Andrew Tomkins. Geographic routing in social networks. Proceedings of the National Academy of Sciences of the United States of America, 102:11623–11628, 2005.

[23] Jacob Eisenstein, Brendan O'Connor, Noah A. Smith, and Eric P. Xing. A latent variable model for geographic lexical variation. In EMNLP '10, pages 1277–1287.

[24] Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. The Journal of Machine Learning Research, 4:933–969.

[25] Gabriel Pui Cheong Fung, Jeffrey Xu Yu, Huan Liu, and Philip S. Yu. Time-dependent event hierarchy construction. In KDD '07, pages 300–309, 2007.

[26] Gabriel Pui Cheong Fung, Jeffrey Xu Yu, Philip S. Yu, and Hongjun Lu. Parameter free bursty events detection in text streams. In VLDB '05, pages 181–192.

[27] Mike Gartrell, Ulrich Paquet, and Ralf Herbrich. A bayesian treatment of social links in recommender systems. In CU Technical Report CU-CS-1092-12, 2012.

[28] Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. Pattern Analysis and Machine Intelligence, IEEE Transactions on, (6):721–741, 1984.

[29] 18 months of all geo-tagged tweets from Twitter firehose. https://www.mapbox.com/blog/visualizing-3-billion-tweets/.

[30] Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. Part-of-speech tagging for twitter: annotation, features, and experiments. In HLT '11, pages 42–47.

[31] Manish Gupta, Jing Gao, Chengxiang Zhai, and Jiawei Han. Predicting future popularity trend of events in microblogging platforms. In Proceedings of the American Society for Information Science and Technology, volume 49, pages 1–10.

[32] John Hannon, Mike Bennett, and Barry Smyth. Recommending twitter users to follow using content and collaborative filtering approaches. In RecSys '10, pages 199–206.

[33] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In ICDM'08, pages 263–272.

[34] Junming Huang, Xue-Qi Cheng, Jiafeng Guo, Hua-Wei Shen, and Kun Yang. Social recommendation with interpersonal influence. In ECAI '10, pages 601–606.

[35] Alexander Ihler, Jon Hutchins, and Padhraic Smyth. Adaptive event detection with time-varying poisson processes. In KDD '06, pages 207–216.

[36] Mohsen Jamali and Martin Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In RecSys '10, pages 135–142.

[37] Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. Why we twitter: understanding microblogging usage and communities. In WebKDD/SNA-KDD '07, pages 56–65.

[38] Meng Jiang, Peng Cui, Rui Liu, Qiang Yang, Fei Wang, Wenwu Zhu, and Shiqiang Yang. Social contextual recommendation. In CIKM '12, pages 45–54.

[39] Ashraf Khalil and Kay Connelly. Context-aware telephony: privacy preferences and sharing patterns. In CSCW '06, pages 469–478.

[40] Jon Kleinberg. Bursty and hierarchical structure in streams. In KDD '02, pages 91–101.

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

[42] Vassilis Kostakos, Jayant Venkatanathan, Bernardo Reynolds, Norman Sadeh, Eran Toch, Siraj A. Shaikh, and Simon Jones. Who's your best friend?: targeted privacy attacks in location-sharing social networks. In UbiComp '11, pages 177–186.

[43] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In WWW '10, pages 591–600.

[44] Vasileios Lampos, Tijl De Bie, and Nello Cristianini. Flu detector-tracking epidemics on twitter. In Machine Learning and Knowledge Discovery in Databases, pages 599–602. 2010.

[45] Theodoros Lappas, Benjamin Arai, Manolis Platakis, Dimitrios Kotsakos, and Dimitrios Gunopulos. On burstiness-aware search for document sequences. In KDD '09, pages 477–486.

[46] Jure Leskovec, Lars Backstrom, and Jon Kleinberg. Meme-tracking and the dynamics of the news cycle. In KDD '09, pages 497–506, 2009.

[47] Cindy Xide Lin, Bo Zhao, Qiaozhu Mei, and Jiawei Han. Pet: a statistical model for popular events tracking in social communities. In KDD '10, pages 929–938.

[48] Tie-Yan Liu. Learning to rank for information retrieval. Foundations and Trends in Information Retrieval, 3(3):225–331, March 2009.

[49] Hao Ma, Irwin King, and Michael R Lyu. Learning to recommend with social trust ensemble. In SIGIR '09, pages 203–210.

[50] Hao Ma, Haixuan Yang, Michael R Lyu, and Irwin King. Sorec: Social recommendation using probabilistic matrix factorization. In CIKM '08, pages 931–940.

[51] Qiaozhu Mei, Chao Liu, Hang Su, and ChengXiang Zhai. A probabilistic approach to spatiotemporal theme pattern mining on weblogs. In WWW '06, pages 533–542.

[52] D. Metzler, S. Dumais, and C. Meek. Similarity measures for short segments of text. Lecture Notes in Computer Science, 4425:16, 2007.

[53] Alan Mislove, Bimal Viswanath, Krishna P. Gummadi, and Peter Druschel. You are who you know: inferring user profiles in online social networks. In WSDM '10, pages 251–260.

[54] Michael C. Mozer, Benjamin Link, and Harold Pashler. An unsupervised decontamination procedure for improving the reliability of human judgments. In NIPS '11, pages 1791–1799.

[55] Ramesh Nallapati, Ao Feng, Fuchun Peng, and James Allan. Event threading within news topics. In CIKM '04, pages 446–453, 2004.

[56] Saša Petrović, Miles Osborne, and Victor Lavrenko. Streaming first story detection with application to twitter. In HLT '10, pages 181–189.

[57] Dragomir R. Radev, Hongyan Jing, Malgorzata Stys, and Daniel Tam. Centroid-based summarization of multiple documents. Information Processing & Management, 40(6):919 – 938, 2004.

[58] RankLib. http://people.cs.umass.edu/~vdang/ranklib.html.

[59] Wolfgang Mueller Raphael Volz, Joachim Kleb. Towards ontology based disambiguation of geographical identifiers. In WWW '07, pages 8–12.

[60] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In WWW '10, pages 851–860.

[61] Ruslan Salakhutdinov and Andriy Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In ICML '08, pages 880–887.

[62] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. Advances in neural information processing systems, 20:1257–1264, 2008.

[63] Gerard Salton, Edward A. Fox, and Harry Wu. Extended boolean information retrieval. Commun. ACM, 26(11):1022–1036, 1983.

[64] Kate Starbird and Leysia Palen. (how) will the revolution be retweeted?: information diffusion and the 2011 egyptian uprising. In CSCW '12, pages 7–16.

[65] Kate Starbird, Leysia Palen, Amanda L. Hughes, and Sarah Vieweg. Chatter on the red: what hazards threat reveals about the social life of microblogged information. In CSCW '10, pages 241–250, 2010.

[66] I. Subasic and B. Berendt. Web mining for understanding stories through graph visualisation. In ICDM '08, pages 570 –579.

[67] Over 4.5 billion check-ins on Foursquare. https://foursquare.com/about.

[68] 340 million tweets per day. http://thenextweb.com/.

[69] Twitter. http://www.twitter.com.

[70] Twitter. Following rules and best practices. https://support.twitter.com/articles/68916-following-rules-and-best-practices, 2013.

[71] Xuanhui Wang, ChengXiang Zhai, Xiao Hu, and Richard Sproat. Mining correlated bursty topic patterns from coordinated text streams. In KDD '07, pages 784–793.

[72] William Edward Webber. Measurement in information retrieval evaluation. 2011.

[73] Jianshu Weng and Bu-Sung Lee. Event detection in twitter. In Proceedings of the 5th International AAAI Conference on Weblogs and Social Media, volume 3, 2011.

[74] Dwi H. Widyantoro, Thomas R. Ioerger, and John Yen. An incremental approach to building a cluster hierarchy. In ICDM '02, page 705, 2002.

[75] Benjamin P. Wing and Jason Baldridge. Simple supervised document geolocation with geodesic grids. In HLT '11, pages 955–964.

[76] Yiming Yang, Tom Ault, Thomas Pierce, and Charles W Lattimer. Improving text categorization methods for event tracking. In SIGIR '00, pages 65–72.

[77] Yiming Yang, Tom Pierce, and Jaime Carbonell. A study of retrospective and on-line event detection. In SIGIR '98, pages 28–36.

[78] Mao Ye, Peifeng Yin, Wang-Chien Lee, and Dik-Lun Lee. Exploiting geographical influence for collaborative point-of-interest recommendation. In SIGIR '11, pages 325–334.

[79] Zhijun Yin, Liangliang Cao, Jiawei Han, Chengxiang Zhai, and Thomas Huang. Geographical topic discovery and comparison. In WWW '11, pages 247–256.

[80] Dejin Zhao and Mary Beth Rosson. How and why people twitter: the role that micro-blogging plays in informal communication at work. In GROUP '09, pages 243–252, 2009.