MINING, MODELING, AND ANALYZING REAL-TIME SOCIAL TRAILS

A Dissertation

by

KRISHNA YESHWANTH KAMATH

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	James Caverlee
Committee Members,	Richard Furuta
	Ricardo Gutierrez-Osuna
	Patrick Burkart
Head of Department,	Hank Walker

August 2013

Major Subject: Computer Science and Engineering

ABSTRACT

Real-time social systems are the fastest growing phenomena on the web, enabling millions of users to generate, share, and consume content on a massive scale. These systems are manifestations of a larger trend toward the global sharing of the real-time interests, affiliations, and activities of everyday users and demand new computational approaches for monitoring, analyzing, and distilling information from the *prospective web* of real-time content.

In this dissertation research, we focus on the real-time *social trails* that reflect the digital footprints of crowds of real-time web users in response to real-world events or online phenomena. These digital footprints correspond to the artifacts strewn across the real-time web like posting of messages to Twitter or Facebook; the creation, sharing, and viewing of videos on websites like YouTube; and so on. While access to social trails could benefit many domains there is a significant research gap toward discovering, modeling, and leveraging these social trails. Hence, this dissertation research makes three contributions:

- The first contribution of this dissertation research is a suite of efficient techniques for discovering non-trivial social trails from large-scale real-time social systems. We first develop a communication-based method using temporal graphs for discovering social trails on a stream of conversations from social messaging systems like instant messages, emails, Twitter directed or @ messages, SMS, etc. and then develop a content-based method using locality sensitive hashing for discovering content based social trails on a stream of text messages like Tweet stream, stream of Facebook messages, YouTube comments, etc.

- The second contribution of this dissertation research is a framework for model-

ing and predicting the spatio-temporal dynamics of social trails. In particular, we develop a probabilistic model that synthesizes two conflicting hypotheses about the nature of online information spread: (i) the spatial influence model, which asserts that social trails propagates to locations that are close by; and (ii) the community affinity influence model, which asserts that social trail propagates between locations that are culturally connected, even if they are distant.

- The third contribution of this dissertation research is a set of methods for social trail analytics and leveraging social trails for prognostic applications like real-time content recommendation, personalized advertising, and so on. We first analyze geo-spatial social trails of hashtags from Twitter, investigate their spatio-temporal dynamics and then use this analysis to develop a framework for recommending hashtags. Finally, we address the challenge of classifying social trails efficiently on real-time social systems.

DEDICATION

Amma, Aanu and Minka

ACKNOWLEDGEMENTS

I would like to express my thanks and gratitude to all who supported and helped me through writing this dissertation. I would also like to thank all my committee members for their advice and support during my Ph.D.

I am deeply indebted to my advisor, Dr. James Caverlee for his unlimited help throughout my research. When I started working with him in August 2008, I didn't really have any research experience. Still he gave me an opportunity to work with him, was patient with me when I stumbled and guided me through the Ph.D process. I am thankful for his support during my hard times and for teaching me valuable life lessons of how to positively cope with failures and rejections. He was not only a good advisor but also a wonderful person, full of cheer, that made my grad school experience memorable. During one of our conversations, I remember him saying that, one of the reasons he became a teacher was so that he could guide students and have an impact on their lives. I think he has succeeded in his goal at least with me. I can undoubtedly say that, after my parents, Dr. Caverlee has been the single biggest influence in my life and professional development. I will forever be grateful for his mentorship and everything he has done for me.

Many thanks to my lab-mates at Infolab for their support and helpful discussions. Senior grad students, Said Kashoob and Elham Khabiri, welcomed me to the lab and helped me understand what it takes to succeed as a graduate student. Jeremy Kelly had a tremendous impact on my PhD. He introduced me to python and map-reduce, two tools which turned out to be integral part of my research. My other lab mates Brian David Eoff, Kyumin Lee and Zhiyuan Cheng with whom I had wonderful discussions, exchange of ideas and collaboration on several interesting projects. My

office mates over the past few years, Yuan Liang and Jeff McGee, who made the experience of grad school very joyful. I am thankful to my roommate of the past 5 years, Ashwin Rao, who was there as a friend and someone who I could always bounce ideas off. I would also like to thank Amayika Panda who made me start thinking about grad school seriously and helped me during the grad school application process.

I wish to thank my parents for their love and support. I always thought my sister and I were lucky to have parents who were encouraging and excellent role models. Things I saw and learned from them like honesty and how they set goals and strived to achieve them has helped me throughout my life especially in the last 5 years. I also wish to thank my sister who knows me the best and has been massively supportive during this time. My family has always trusted me and my decision making, which has given me the confidence to take risks and go after my goals. My parents valued good education more than anything else and made sure we had the best environment for it growing up. This along with the sacrifices they made are the main reasons why I am here today. Thanks for all the things you have done for me. Love you all.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

1.1 Motivation

Over the past few years we have seen an exponential growth in social media. This growth has been fueled by advancements in two complementary areas: (i) proliferation of devices with Internet access like hand-held devices (smart phones, tablets), smart TVs, and gaming consoles; and (ii) the growth of several content delivery services (YouTube, Netflix, iTunes) and real-time social systems for information sharing (Twitter, Facebook, Reddit). These advances have not only ensured that people spend more time on the Internet but also have given users new ways to interact with the online content and give their feedback. Increased interaction among users and content has allowed us to collect their social trails (digital footprints) both explicit – like tweets, and Facebook likes – and implicit – like query logs and click-through logs – at scale that was not possible a few years ago.

Consider the rapid evolution of the social web over the last decade. The web started gaining traction with the introduction of online social networks (e.g. Facebook, Myspace, Orkut), that allowed friends to connect with each other. Facebook, today – just over 8 years since its creation – counts one sixth of the world's population in its monthly user base, of which more than 604 million users visit it from their mobile phones [25]. Alongside the popularity of social networks came multi-media sharing services (Youtube, Flikr) which allowed users to share videos, pictures, and other media with other users of the service. Youtube, started in 2005, currently sees more than 4 billion video hits per day [58]. Blogging websites (Blogger, Wordpress) were followed by micro-blogging services (Twitter, Plurk) that allowed users to post short messages. Though the first tweet was posted in 2006, today Twitter generates more than 200 million tweets every day [24]. More recently we have seen the rise of location based social networks (Foursquare, Google Latitude) that allows users to share their location with friends. Foursquare in five years has accumulated more than 3 billion checkins (geo-impressions) with millions of new checkins added every day [28]. In this way, rapidly evolving social services with their ever increasing user base are generating large-scale digital footprints which can be leveraged to build interesting data-driven applications.

Hence, in this dissertation research, we focus on the real-time *social trails* that reflect the digital footprints of crowds of real-time web users in response to real-world events or online phenomena. These digital footprints correspond to the artifacts strewn across the real-time web. Common examples include the posting of messages to Twitter or Facebook; the creation, sharing, and viewing of videos on websites like YouTube; and the revelation of user locations through location-sharing services like Foursquare and Google Latitude. Together these social trails embody the online evolution of crowds of real-time web users.

Discovery, modeling and analyzing social trails could benefit many domains. From the early days of search engines, companies have been using implicit trails in the form of query logs to understand and improve their webpage ranking algorithms. Google has also used query logs to understand popularity of various concepts and use it to predict trends [15]. Another application of social trails is in the epidemiological and disease control domain, where experts could search them for evidence of new outbreaks and the reaction of the public to new vaccines. An example of such an application is Google flu trends [35], that combines good indicators of flu in search terms to track the epidemic. Social trails can also be used in other purposes like municipalities interested in responding to local events (like the recent Vancouver riots), finance experts monitoring stock price jumps or crashes, political scientists tracking chatter about presidential debates and so on.

Just as web search engines provide instant access to the *retrospective web* of previously crawled and indexed content, the real-time social systems, that form the back bone of the social web, demand new computational approaches for monitoring, analyzing, and distilling information from the *prospective web* of real-time content that reflects the current (and future) activity of web users.

1.2 Research Challenges

In the previous section, we described social trails and why they are important. We now identify some of the research challenges associated with social trails. To satisfy the potential that social trail analysis holds there are significant research gaps toward discovering, modeling, and leveraging these social trails. For example:

- **Real-time Nature and Large Scale**: Most existing web mining techniques are ill-suited for the challenges inherent in discovering real-time social trails. For example, existing techniques like map-reduce are designed to handle large datasets but are inefficient when applied on large scale information streams to produce results in real-time.

- **Unknown Properties of Social Trails**: There is little understanding of the properties of social trails. For example, what are the spatio-temporal dynamics of social trail evolution? What impacts the growth or fall of social trails? Analyzing the properties of social trails is very important while developing prognostic applications like recommendation engines, advertisement targeting, and so on.

- **Lack of Analytics**: Due to absence of real-time analytics to quantify social trails, there is a lack of understanding of the types of applications that can leverage social trails and the impact of this leverage on application performance. For example, can we incorporate artifacts from social trails in web-search ranking and is there a performance improvement observed because of this?

1.3 Contributions

With these research challenges in mind, this dissertation seeks to develop new algorithms and methods to discover, model and analyze social trails on the real-time web. Concretely, this dissertation takes a three-fold approach:

1.3.1 Part I: Social Trails Discovery

The first contribution of this dissertation research is a suite of efficient techniques for discovering social trails from large-scale real-time social systems. We view a social trail as an evolving set of *transient crowds* and focus on the task of first extracting these transient crowds. Each transient crowd (or just crowd) is a potentially short-lived ad-hoc collection of users (and their associated content) at the core of a social trail that triggers its formation and contributes to its evolution. Concretely, we first develop a communication-based method using temporal graphs for discovering social trails on a stream of conversations from social messaging systems like instant messages, emails, Twitter directed or @ messages, SMS, etc.

We then develop a content-based method using locality sensitive hashing for discovering content based social trails on a stream of text messages like Tweet stream, stream of Facebook messages or YouTube comments. We evaluate the performance of our social trail discovery algorithms over Twitter datasets and through extensive experimental study, we find our algorithms to be efficient while maintaining high-quality crowds as compared to other approaches.

1.3.2 Part II: Social Trails Modeling

The second contribution of this dissertation research is a framework for modeling and predicting the spatio-temporal dynamics of social trails. By modeling trail propagations we want to answer questions like, how did social trails of videos captured on smart-phones during the Arab Spring spread across the globe? Are there key locations that promoted the propagation of these trails? As the Arab Spring became increasingly part of the US's social conscious, did we see key US locations impacting the propagation of these trails that were not influential in the past?

In particular, we develop a probabilistic model that synthesizes two conflicting hypotheses about the nature of online information spread: (i) the spatial influence model, which asserts that social trails propagates to locations that are close by; and (ii) the community affinity influence model, which asserts that social trail propagates between locations that are culturally connected, even if they are distant. We test these models in the context of the geospatial footprint of 755 million geo-tagged hashtags and find that while the spatial influence model had a higher impact than the community affinity influence model in predicting the spread, its combination with community affinity influence model gave the best performance, suggesting that both distance and community are key contributors to social media spread. The combination of these models is able to predict flow close to 80% accuracy of the best possible model.

1.3.3 Part III: Social Trails Analytics

The third contribution of this dissertation research is a set of methods for social trail analytics and leveraging social trails for prognostic applications like real-time content recommendation, personalized advertising, and so on. We first analyze geo-spatial social trails of hashtags from Twitter and investigate their spatio-temporal dynamics. Our investigation is structured in three steps. First, we study the global footprint of hashtags and explore the spatial constraints on hashtag adoption. Second, we study three spatial properties of hashtag propagation – focus, entropy, and spread – and examine the spatial propagation of hashtags using these properties. Finally, we present two methods for characterizing locations based on hashtag spatial analytics. Based on the insights we gain during modeling social trails and and their geo-spatial properties we then address the challenge of classifying social trails efficiently on real-time social systems.

We then present an expert-driven framework for time-aware topical classification framework for social trails. The key insight driving the framework is the reliance on category-specific *experts*, whose Twitter streams themselves may serve as prototypes for learning generalized categorical models for robust trail classification. We show how these expert streams may seed classification, and we propose a sliding-window training approach for adaptive topical classification. Additionally, we explore techniques for augmenting short messages using feature-based, link-based and collocation expansion. Through experimental study over Twitter, we find good performance of the proposed method for ongoing expert-driven topical classification of social trails.

1.4 Dissertation Overview

The remainder of this dissertation is organized into four parts, of which the first three are for our contributions and the fourth for conclusions. The outline is as follows:

- Social Trails Discovery
  - **Section 2: Discovery of Communication Based Social Trails** - We begin with describing an approach to discover social trails in social messaging systems. We propose a message-based communication clustering

approach over time-evolving graphs that captures the natural conversational nature of social messaging systems.

  - **Section 3: Discovery of Content Based Social Trails** - We propose and evaluate a novel content-driven social trail discovery algorithm that can efficiently identify newly-formed communities of users from the real-time web. Three of the salient features of the algorithm are its: (i) prefix-tree based locality-sensitive hashing approach for discovering trails from high-volume rapidly-evolving social media; (ii) efficient user profile updating for incorporating new user activities and fading older ones; and (iii) key dimension identification, so that trail detection can be focused on the most active portions of the real-time web.

- Social Trails Modeling

  - **Section 5: Modeling of Geo Based Social Trails** - We seek to understand and model the global spread of social trails. We develop a probabilistic model that synthesizes two conflicting hypotheses about the nature of online information spread: (i) the spatial influence model, which asserts that social media spreads to locations that are close by; and (ii) the community affinity influence model, which asserts that social media spreads between locations that are culturally connected, even if they are distant.

  - **Section 4: Analysis of Geo Based Social Trails** - We conduct a study of the spatio-temporal dynamics of social trails (Twitter hashtags) through a sample of 2 billion geo-tagged tweets. In our analysis, we (i) examine the impact of location, time, and distance on the adoption of hashtags, which is important for understanding meme diffusion and information propagation; and (ii) examine the spatial propagation of hashtags through their focus, entropy, and spread;

- Social Trails Analytics
  - **Section 6: Real-time Recommendation of Social Trails** - Based on the analysis of previous two sections, in this section we develop techniques that can be used to recommend social trails that will be popular at any location. We develop feature functions to predict expected growth of social trails at a location. We then use machine learning algorithms to learn the best feature function or the best combination of feature function for a particular location.
  - **Section 7: Real-time Classification of Social Trails** - We study the problem of expert-driven topical classification of social trails in time-evolving streams like Facebook status updates, Twitter messages, and SMS communication. Three of the salient features of the framework are (i) a novel expert-centric classifier; (ii) a sliding-window training for adaptive topical classification; and (iii) a suite of enrichment-based methods (lexical, link, collocation) for overcoming feature sparsity in short messages.
  - **Section 8: Visualization of Locations Using Geo Based Social Trails** - We present two methods for characterizing locations based on hashtag spatial analytics. The first method uses spatial properties – entropy and focus – to determine the nature of a location from the point of hashtag propagation using location-entropy-focus-spread plots, while the second method uses hashtag adoption times to characterize a location's impact to enable hashtag propagation.

- Conclusion
    - **Section 9: Summary and Future Research Oppurtunities** - We conclude with a summary of our dissertation contributions and a discussion of future research extensions to the results presented here.

2. DISCOVERY OF COMMUNICATION BASED SOCIAL TRAILS*

In this section we describe our approach to discover social trails from large scale real-time social streams. We view a social trail as an evolving set of transient crowds and focus on the task of first extracting these transient crowds. Each transient crowd (or just crowd) is a potentially short-lived ad-hoc collection of users (and their associated content) at the core of a social trail that triggers its formation and contributes to its evolution. In general, a crowd could be defined by the posting and sharing actions of users in social systems, for example triggered by an offline event (e.g., Facebook posts and Tweets in response to a live Presidential debate or a chemical fire at a nearby refinery) or by an online phenomenon (e.g., reaction to Internet memes, online discussion).

2.1 Introduction

Transient crowds could be viewed through several overlapping perspectives: (i) *communication-based*, reflecting groups of users who are actively messaging each other, e.g., users coordinating a meeting; (ii) *location-based*, reflecting groups of users who are geographically bounded, e.g., users posting messages from Houston, Texas; and (iii) *interest-based*, reflecting groups of users who share a common interest, e.g., users posting messages about a presidential debate. In this section, we focus on discovery of communication-based crowds.

Transient crowds are dynamically formed and potentially short-lived. Hence, it is a major challenge to efficiently identify coherent crowds across a potentially vast collection of non-obviously connected user actions. Considering Twitter alone, there

*Part of this section is reprinted with permission from "Transient Crowd Discovery on the Real-Time Social Web" by Krishna Y. Kamath and James Caverlee, 2011. Web Search and Data Mining. 4th. Copyright 2013 by Association for Computing Machinery (ACM).

are potentially 100s of millions of active users inserting new messages into the system at a high-rate. How can we identify and extract real-time crowds efficiently without sacrificing crowd quality? In addition to identifying a particular crowd at a point-in-time, how can we efficiently and successfully track the crowd over time as users join, crowds merge, and crowds disperse?

We propose to model crowd formation and dispersion through a message-based communication clustering approach over time-evolving graphs. Two of the salient features of the proposed approach are (i) an efficient locality-based clustering approach for identifying crowds of users, and (ii) a novel crowd tracking and evolution approach for linking crowds across time periods. The efficient locality-based clustering is developed on the notions that (i) changes in a small region of a graph should not affect the entire graph; and (ii) that edge weights should reflect temporal and interest locality (e.g., decaying based on communication recency).

2.2 Problem Statement

We are interested in exploring short-lived group formations on large and growing social messaging systems like Twitter and Facebook. As we have noted, users on these social networks may be grouped along a number of dimensions including content-based (or thematic interest), geographic-based, communication-based, and so on. In this section, we focus on the specific challenge of uncovering and tracking groups of users – what we refer to as *transient crowds* – according to their communication patterns. Compared to previous works [54, 71] that seek to do fast clustering on an as-needed basis, our approach is to detect and track crowds in real-time (e.g., every minute). This requires both a single-shot fast clustering and cluster evolution to track changes and trends. In addition these previous works use offline algorithms which are not suitable for our requirements.

Historically, direct communication between people has been mostly unobservable or unavailable for large-scale web mining. For example, private email and instant messages between two users are typically not made available for natural reasons. But with the rise of new social messaging systems like Twitter and Facebook, communications between users can be monitored. For example, Twitter supports the public messaging of users through the inclusion of @$\langle$*username*$\rangle$ in a Twitter post (a "tweet"). So a tweet from the user *nod* can be addressed to the user *kykamath* like so: "@kykamath What do you think about the new iPad?". This type of observable communication is on the rise and is a significant portion of all messages posted on Twitter, with estimates placing the percent of all tweets containing the @$\langle$*username*$\rangle$ at 30% (or about 7 million observable communications per day). Similar messaging functionality has recently been adopted by Facebook. Based on these observable communication patterns, we study how to efficiently discover and track transient crowds. We now give some definitions before framing the problem.

Definition (Time-Evolving Communication Network) A time-evolving communication network is an undirected graph $G_t(V,\ E)$ graph with $|V| = n$ vertices and $|E| = m$ edges, where each vertex corresponds to a user in the social messaging system and an edge corresponds to a communication between two users. The weight of an edge between vertices u and v at time t is represented by $w_t(u,\ v)$.

The communication network is time evolving because the relationship between users – as indicated by $w_t(u,\ v)$ – changes over time. In practice, the edge weights in a time-evolving communication network could be based on the geographical distance between users, the "semantic" closeness based on an analysis of the content of their messages, or other context-sensitive factors. For concreteness, in this study we focus on purely communication-based properties (the recency and number of messages between the users) for determining the edge weights in the time-evolving communication network.

Definition (Transient Crowd): A transient crowd $C \in K_t$ is a time-sensitive collection of users who form a cluster in G_t, where K_t is the set of all transient crowds in G_t. A transient crowd represents a collection of users who are actively communicating with each other at time t.

Based on these definitions, we can now break our problem into two parts:

(i) *Crowd Discovery Problem*: Discover the set of transient crowds K_t that exist in the communication network $G_t(V,\ E)$ at time t; and

(ii) *Crowd Tracking Problem*: Track the evolution of transient crowds discovered across time periods as they grow, merge, split, and disperse.

2.2.1 Example

To illustrate the problem of crowd discovery, consider the simple example in Figure 2.1. At time t=1, users A and B send messages to each other, as do users C and D.[†] The associated communication graph shows an edge between the two pairs, where for simplicity the edge is annotated with the number of messages between the users (2, in both cases). Further, suppose we identify crowds based purely on graph connectivity. So for time t=1, we see there are two crowds discovered {A,B} and {C,D}. For each crowd, we can characterize the semantics of their communication with simple keywords extracted from the content of the tweets: ("oil", "gulf") and ("walcott", "capello"). At time t=2, the communication graph is updated with a new edge (connecting User A and User C), and the existing edges are decayed by one (again, a simplifying assumption for the purposes of this example). A single crowd

[†] For simplicity, the example discretizes time so that all messages between users occur in steps. In practice, the proposed algorithm relaxes this assumption and can handle arbitrary message sending times.

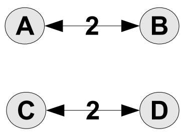
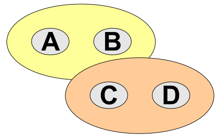
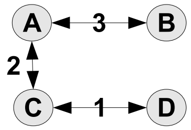
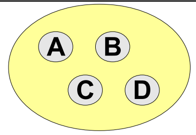
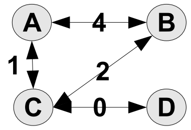
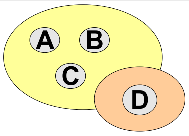
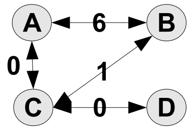
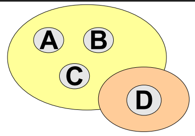
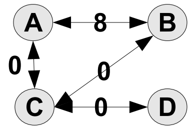
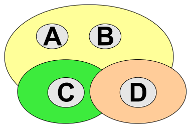
t	Twitter @ messages	Communication Graph	Crowds Discovered	Crowd Analysis
1	A: @B BP modifies Gulf oil cleanup plan. B: @A Feds Open Criminal Probe on Oil. C: @D Fabio Capello's England. D: @C Walcott dropped.			🟨 - bp, gulf, oil, fed 🟧 - walcott, capello
2	A: @B Marine Life dying in Gulf Coast. A: @C Gulf Oil Spill: Diamond saw breaks. C: @A Oil spill protest tomorrow			🟨 - gulf, oil, spill
3	A: @B 10 things to hate about BP. B: @C Huge environmental impact. C: @B Protesting oil spill at NY.			🟨 - bp, protest, environment
4	A: @B Top kill fails. B: @A BP doesn't care.			🟨 - bp, carem top, kill
5	A: @B Hope things get better over weekend. B: @A Deep water will take down BP.			🟨 - deep, water, bp, weekend

Figure 2.1: Example of crowd discovery and tracking in Twitter.

is discovered since all users are connected via edges with non-zero edge weights. At time t=3, User D leaves the main crowd since no messages to or from User D have been observed since time t=1. This process continues until time t=5 when User C also leaves the main crowd due to inactivity. Note that crowds are discovered from communication graph only and not from the content of the messages. As an example of crowd tracking, we can track the evolution of the yellow crowd across time periods, observing the changes it goes through as it grows in size from t=1 to t=2 and then reduces to two users by t=5.

2.2.2 Challenges

Based on the simple example above, we could imagine directly scaling the basic transient crowd discovery and tracking approach to systems like Facebook and Twitter. For practical crowd discovery and tracking in a large time-evolving communication network, however, we face four key challenges:

- First, systems like Facebook and Twitter are extremely large (on the order of 100s of millions of unique users), placing huge demands on the computational cost of traditional community detection approaches (which can be $O(n^3)$ in the number of users [27]).

- Second, these services support a high-rate of edge addition (new messages) so the discovered crowds may become stale quickly, resulting in the need to re-identify all crowds at regular intervals (again, incurring the high cost of community detection). The bursty nature of user communication demands a crowd discovery approach that can capture these highly-temporal based clusters.

- Third, the strength of association between two users may depend on many factors (e.g., recency of communication), meaning that a crowd discovery approach based on graph clustering should carefully consider edge weights. With no decay at all (meaning that edges are only inserted into the network but never removed), all users will tend towards a single trivial large crowd. Conversely, overly aggressive edge decay may inhibit any crowd formation at all (since edges between users may be removed nearly as soon as they are added).

- Fourth, crowds may evolve at different rates, with some evolving over several minutes, and others taking several days. Since crowds are inherently ad-hoc (without unique community identifiers – e.g., Fans of LA Lakers) the formation, growth and dispersal of crowds must be carefully managed for meaningful crowd analysis.

2.3 Crowd Discovery and Tracking

With these challenges in mind, we propose to discover and track transient crowds through a communication-based clustering approach over time-evolving graphs that captures the natural conversational nature of social messaging systems. Two of the salient features of the proposed approach are (i) an efficient locality-based clustering approach for identifying crowds of users in near real-time compared to more heavyweight static clustering algorithms; and (ii) a novel crowd tracking and evolution approach for linking crowds across time periods. In the rest of this section we tackle each of these key areas in turn before evaluating the proposed approach in Section 2.4 (Experiments).

2.3.1 Locality in Social Messaging Systems

To support transient crowd discovery in Twitter-like services with 100s of millions of participants, we propose to leverage the inherent locality in social messaging systems. Concretely, we identify two types of locality that are evident in Twitter-like messaging systems: (i) temporal locality and (ii) spatial locality.

Temporal Locality: Transient crowds are intuitively short-lived, since they correspond to actively communicating groups of users. Hence, the composition of a crowd at a point-in-time should be impacted by recent messages as opposed to older messages. As more users interact with the crowd, the crowd should grow reflecting this *temporal locality* and then shrink as users in the crowd become inactive (that is, their last communication with the crowd becomes more distant in time).

Spatial Locality: Intuitively, transient crowds are made up of a very small percentage of users compared to the entire population of the social network. Hence, new messages (corresponding to the addition of edges to the communication network) should have only a local influence on the crowds that exist at any given time. That is, changes in a small region of a graph should not affect the entire graph. In a dataset of 61 million Twitter messages described in Section 2.4, we have confirmed the existence of this *spatial locality* by finding that only about 1% of users are within two hops, meaning that an edge insertion has only a local effect.

Hence, we can take advantage of both, local changes to the overall communication network (spatial locality) and recent changes to the network (temporal locality), for supporting efficient transient crowd discovery. We next describe how we can use these locality properties in our proposed solution.

2.3.2 Modeling Temporal Locality

Temporal locality suggests that transient crowds should be composed of users who have communicated with the crowd recently, and that older messages should be treated less significantly. In the motivating example in Figure 2.1, we implemented temporal locality by reducing the edge weight by 1 at each time step if no messages are exchanged in a particular time interval, and increasing the edge weight by 1 if messages were exchanged. In the following discussion we explore some more refined approaches for exploiting temporal locality for transient crowd discovery.

Recall that the time-evolving communication network $G_t(V, E)$ has edge weights between vertices u and v at time t represented by $w_t(u,\ v)$. Suppose that the network also stores the latest time any two users communicated $\tau(u,\ v)$ and that we have access to the current time in the system T_{now}.

Fixed window: One approach to model temporal locality is to consider only edges within a fixed time-window β. That is, consider only edges $(u,\ v)$ such that $T_{now} - \tau(u,\ v) < \beta$. In this case, messages sent more than β time units earlier are completely disregarded by the crowd discovery system. A common problem with such a windowing approach is the loss of historical information. In our case this means a possibility that we will miss some significant edges, just because a user didn't communicate in the last β time units. For example, consider 2 users who have constantly exchanged messages over a year, except for the last 1 week. If β is set to 1 week, then the relationship between these 2 users is lost. Hence, using this approach might result in the discovery of imprecise crowds.

Exponential Decay: Alternatively, we propose an edge-weight decay function that gradually fades the impact of older messages relative to newer ones. Concretely, we propose an exponentially decaying impact function based on a decaying coefficient ξ for controlling the rate of decay. The value of ξ determines the type of crowds we identify. Crowds forming slowly can be identified with lower values of ξ while a higher value of ξ identifies only crowds forming quickly. Hence, this parameter can be tuned according to the particular application requirements. Since we are interested in transient crowds we will set the values of ξ to relatively higher values.

For edges $(u,\ v) \mid w(u,v) > 0$ we update the new edge weight, conditioned on message exchange, at time t as:

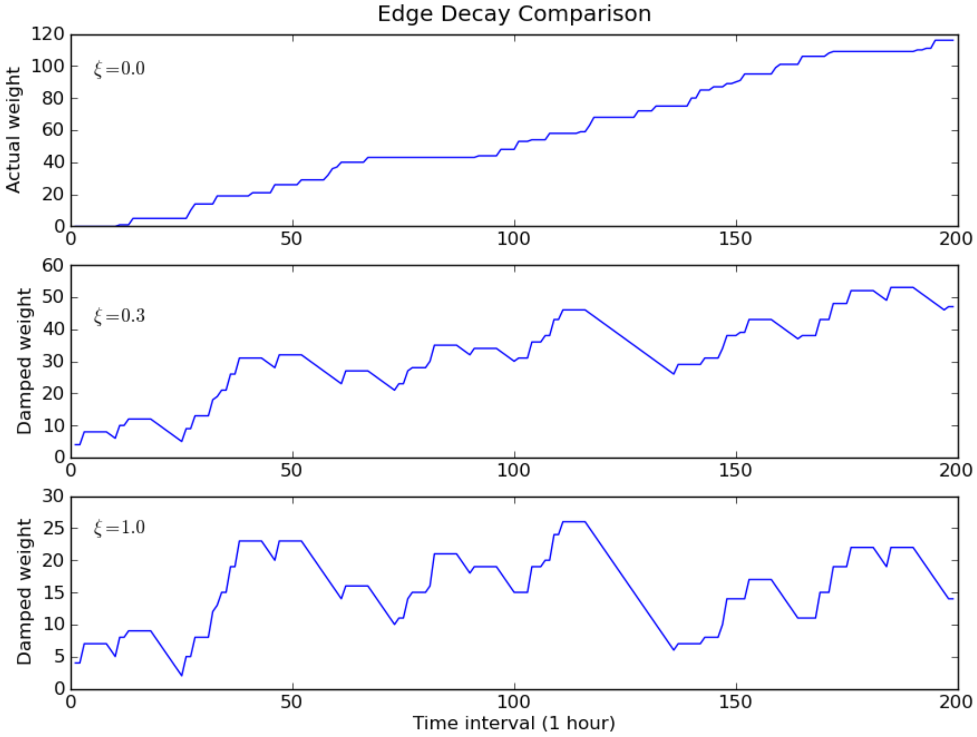


Figure 2.2: Changes to edge weights with exponential decay.

Messages not exchanged:

$$w_t(u,\ v) = w_{t-1}(u,\ v) - \log(T_{now} - \tau(u,\ v))\ \times\ \xi$$

Messages exchanged:

$$w_t(u,\ v) = w_{t-1}(u,\ v) + 1 - \log(T_{now} - \tau(u,\ v))\ \times\ \xi$$

To illustrate the impact of this exponential fading, a typical communication graph between two users is shown in Figure 2.2. The upper plot shows the number of messages exchanged between two users and the bottom plots shows the exponentially decayed edge weights. The middle plot shows the effect of exponential decaying with $\xi = 0.3$ and the bottom plot with $\xi = 1.0$. As expected, we observe that edge weights fall much faster in the bottom plot than in the middle plot.

2.3.3 Exploiting Spatial Locality

Given the temporal locality-inspired optimization of transient crowd discovery, we now turn to spatial locality. To take advantage of spatial locality, we propose to augment a traditional (expensive) graph clustering algorithm by selectively applying the algorithm to small portions of the entire communication network, thereby saving the computational cost of running the algorithm over the entire large network.

Let C_{ti} represent the i^{th} crowd in K_t. Users are assigned to one and only one crowd, i.e., $C_{ti} \cap C_{tj} = \phi, \ \forall \ C_{ti}, C_{tj} \in K_t$. To discover K_t, we could apply one of a number of graph clustering algorithms, including MCL [74], multilevel graph clustering [21], etc. For concreteness, we consider min-cut clustering [27, 33], a popular graph clustering algorithm that has shown good success across real-world datasets like web pages, citation networks, etc. While the following discussion focuses on min-cut clustering (in the interest of providing a baseline for experimental comparison of transient crowd discovery), the general locality principles discussed in this section could be applied to other clustering algorithms.

2.3.3.1 Preliminaries

To begin our development of locality-based clustering, we first present some preliminaries to describe min-cut clustering.

Minimum cut: The minimum cut of a graph G with respect to vertices s and t, where $s \in S, t \in T$, is defined as partition of V into S and T such that, the total weight of edges connecting the partitions is minimum. This is represented as $c(S, T)$.

For an undirected graph G we can define a weighted tree T_G called the minimum-cut tree [33]. We can determine $c(S, T)$ by analyzing the path from s to t in T_G, where the value of $c(S, T)$ is equal to the smallest edge on this path.

Min-cut clustering: The min-cut clustering algorithm [27] clusters a graph G first by adding an artificial sink t. All of the vertices of G are connected to the artificial sink with an edge capacity of α, to form a modified graph G', where α is a parameter guiding the quality guarantees of the resulting clusters. The minimum-cut tree T' for G' is then calculated. The connected components of T' obtained after removing the artificial sink t are clusters in G. Min-cut clustering relies on the special parameter α to ensure the quality of the clusters generated, where:

$$\frac{c(S, V-S)}{|V-S|} \leq \alpha \leq \frac{c(P, Q)}{min(|P|, |Q|)} \tag{2.1}$$

with, $P \cap Q = \phi$ and $P \cup Q = S$. By tuning this α parameter, the number and size of the resulting clusters can be varied (from one large cluster with all nodes to a trivial clustering consisting of n singleton nodes).

2.3.3.2 Locality-Based Clustering

Of course, we could directly apply the min-cut clustering algorithm to the large time-evolving communication network G_t directly. The output would be a set of clusters K_t which we could take to be transient crowds, however, at a considerable expense. Coupled with the need to re-compute clusters as the network evolves, straightforward application of a traditional graph clustering approach is infeasible for efficient transient crowd discovery.

Towards exploiting spatial locality for efficient crowd discovery, we must address two issues: (i) The application of min-cut clustering to a particular subgraph of the entire communication network; and (ii) The determination of which subgraphs of the communication network to select for clustering.

Subgraph clustering: The first challenge is to perform local clustering, given an identified region of the communication network (corresponding to some locally impacted portion of the network). By clustering a local region of the communication network we can begin to reduce the expense of clustering the entire network.

Given a subgraph S (the part of the communication network impacted by edge addition) to cluster, the algorithm first contracts G_t to G'_t. As shown Algorithm 1, this approach then creates a new graph G''_t by adding an artificial sink w_s to G'_t and connecting all the vertices of S to t with edges of capacity α and all the vertices in $(V' - S)$ with edges of capacity of $\alpha|V - S|$ as in [65]. It then determines the minimum-cut tree T''_t for G''_t. The connected components obtained after removing w_s from T''_t are the new clusters (which correspond to transient crowds). In this way, only a small portion of the communication network is impacted, leading to more efficient clustering that clustering the entire network.

Algorithm 1 ClusterSubGraph(S)

1. Contract G_t**:** Reduce G_t to G'_t by replacing vertices $V - S$ with a new vertex x. All the resulting loops are deleted and parallel edges are replaced with a single edge with weight equal to the sum of the edges.

2. Expand G'_t**:** Construct a new graph G''_t by adding vertex w_s to $G'_t(V', E')$. Connect w_s to $v, \forall v \in S$ with edge capacity of α and w_s to $v', \forall v' \in (V' - S)$ with edge capacity of $\alpha|V - S|$.

3. Minimum-cut tree: Determine minimum-cut tree T''_t for G''_t. The connected components obtained in T''_t after removing vertices w_s and x from it are the clusters in S.

Subgraph selection: The second challenge is to determine which subgraphs are to be selected for clustering, i.e. how do we select S in Algorithm 1? Selecting too many subgraphs for re-clustering may result in expensive computation, whereas selecting too few may result in poor crowd quality. Following [65], we adopt an approach triggered on *each edge insertion* to identify subgraphs that need to be clustered.

Depending on the position where an edge is inserted and the effect of edge addition on the quality of clustering there are four ways to select clusters for local clustering. The first case is when an edge is added within an existing cluster C_u. In this case there is a probability that this addition might have resulted in subclusters within C_u, that improve clustering quality. Hence, only C_u is selected for clustering ($Case\ i$). An edge can also be added between 2 clusters. In this case, if the quality of clustering is maintained in spite of this edge addition, then re-clustering is not required ($Case\ ii$). Otherwise, if the quality of clustering is reduced, then we select both clusters for re-clustering ($Case\ iv$). If the addition of an edge between 2 clusters results in satisfying the condition for cluster merging, then the 2 clusters are merged ($Case\ iii$). The pseudocode for subgraph selection is given in *Step 2* of Algorithm 2.

The proof of correctness of these cluster selection methods is given in [65]. We empirically validate the clustering quality in Section 2.4.

Time Complexity Analysis: The algorithm to cluster subgraphs uses the relabel-to-front approach of the push-relabel algorithm [30] to calculate the minimum-cut tree. It has a time complexity of $O(l^3)$, where l is the number of vertices in the minimum-cut tree. Let $k = \max_{i=1}^{|K_t|}(|C_{ti}|)$, the size of the largest crowd in K_t. In edge addition algorithm when both the vertices of the edge belong to the same crowd we decay $O(k^2)$ edges and re-cluster $O(k)$ vertices. In this case the time complexity is $O(k^3)$. In case where the quality of crowds is maintained on addition of the edge, the time complexity is $O(2k^2)$ for damping the edges. During the merge

Algorithm 2 Locality Clustering Algorithm

For every new edge (u, v) added to the graph, perform the following 3 steps.

1. Initialization: If the added edge has vertices that have not been observed before add them to vertex set V. Create singleton clusters for the new vertices and add them to cluster set K.

2. Clustering: Let u, v belong to clusters C_u and C_v respectively. For every edge (internal and boundary) in C_u and C_v decay the edge weights as mentioned in Section 2.3.2. Now depending on the conditions that match perform the corresponding clustering operations:

Case i. If the vertices belong to same cluster then updating the edge weights might have resulted in formation of clusters within this cluster. Check for new clusters using $ClusterSubGraph(C_u)$.

Case ii. If the vertices belong to different clusters and the addition of the edge does not reduce the quality of clustering, then perform no action. The quality of the clustering is maintained if the following inequalities (Equation 2.1) are satisfied.

$$\frac{c(C_u, V-C_u)}{|V-C_u|} \leq \alpha \text{ and } \frac{c(C_v, V-C_v)}{|V-C_v|} \leq \alpha$$

Case iii. If the vertices belong to different clusters and the addition of the edge satisfies the merging condition $\frac{2c(C_u,C_v)}{|V|} \geq \alpha$, merge the 2 clusters.

Case iv. If the vertices belong to different clusters and the previous 2 conditions are not met then the quality of clustering has reduced. Hence, perform $ClusterSubGraph(C_u \cup C_v)$ to generate clusters that maintain clustering quality.

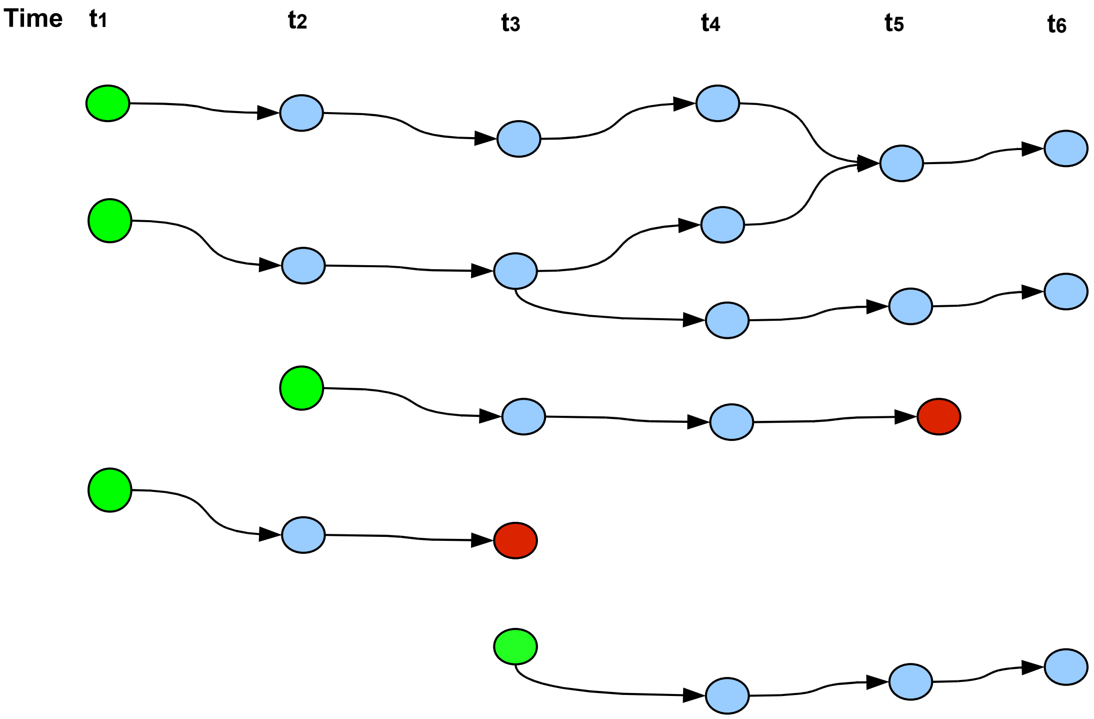


Figure 2.3: Crowd tracking graph. The green nodes represent start of a crowd and the red node shows the dispersing of the crowd.

operation, we dampen $O(2k^2)$ edges and re-cluster $O(2k+1)$ vertices, which results in a time complexity of $O(k^3)$. Hence, the time complexity of the algorithm on an edge addition is $O(k^3)$, as compared to the time complexity $O(n^3)$ for the original min-cut clustering algorithm in [27].

To summarize, in this section we have described how we can use the spatial and temporal locality observed in social messaging systems to design an efficient clustering algorithm.

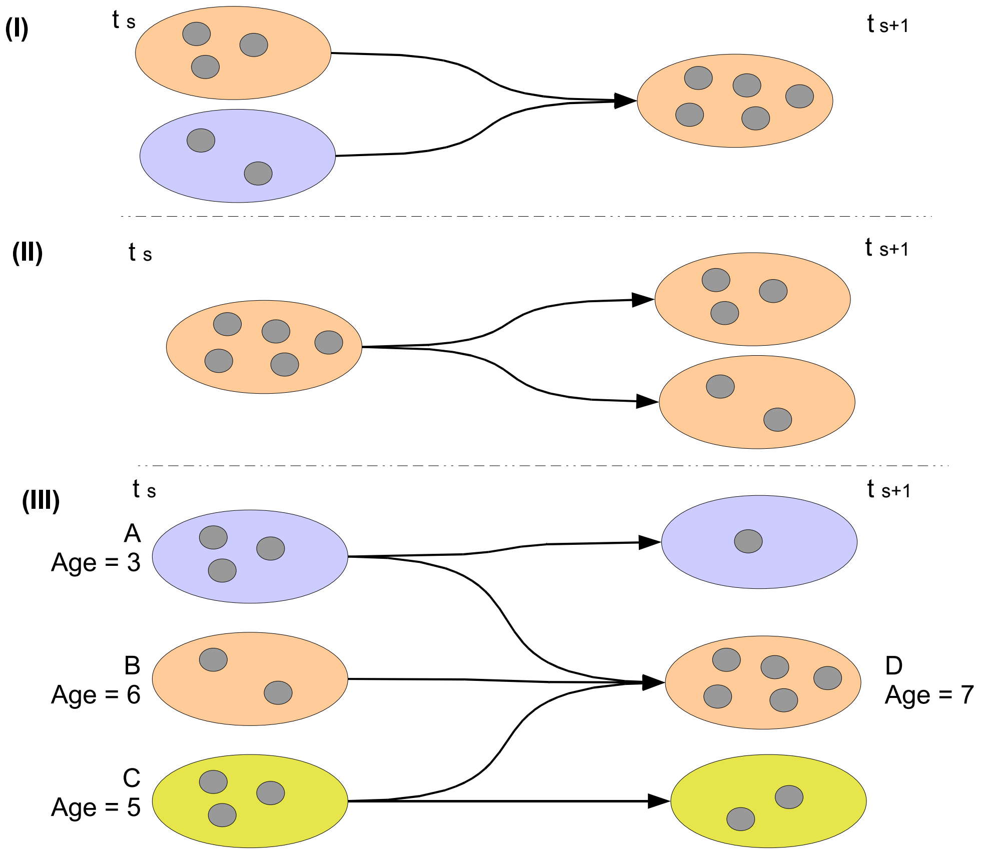


Figure 2.4: Examples of crowd modification events. Arrows indicate crowds in t_s that contribute to crowds in t_{s+1}. Color of the crowd in t_{s+1} indicates its parent in t_s.

2.3.4 Crowd Tracking

Finally, we turn to the second of two key challenges for transient crowd discovery and tracking – how to track crowds over time as users join, crowds merge, and crowds disperse. For example, when we discover a new crowd that is discussing an upcoming event (say the World Cup), we need a method to track the users participating in this crowd in consequent intervals. This would give us an ability to analyze crowd dynamics leading up to and after the event.

Recently, there has been some work analyzing communities across times. In

[6], the authors look at communities on large social networks like LiveJournal and MySpace. Since the communities are explicitly defined in these networks, the task of determining evolution of graph is trivial. In [3] the authors observe changes clusters undergo between time intervals and consider the changes to be events.

Crowd tracking would be straightforward if each crowd were associated with a unique community identifier (e.g., Fans of LA Lakers). Facebook and Twitter have adopted methods for group affiliation like fanclubs and lists, but these longer-lived affiliations are not available nor appropriate for short-lived transient crowds. Since crowds are inherently ad-hoc we define in this section the problem of crowd tracking and present a graph-based approach to solve it.

Crowd Tracking Graphs: A crowd tracking graph G_c is constructed using the crowds obtained at different time intervals. This graph helps us understand the changes that take place in these crowds between time intervals. It is a directed graph with crowds as vertices and the direction of the edge denoting the parent-child relationship. Node colors are used to indicate the state of crowd evolution. A green node indicates that the crowd has been discovered for the first time and a red node indicates dispersal of the crowd. Intermittent crowds are shown in blue color. To track the evolution of a crowd we start at the green node and follow the edges until we reach the red node. An example of a crowd tracking graph is shown in Figure 2.3. The graph also shows examples of merging and splitting of crowds.

Transient Crowd Tracking Problem: Given a time-evolving communication network $G_t(V,\ E)$ and a set of transient crowds K_t identified at every time interval t, construct a Crowd Tracking Graph G_c.

We propose an algorithm to construct a crowd tracking graph. The algorithm takes the crowd set for the s^{th} interval K_s and the crowd set for previous interval

Algorithm 3 Crowd Tracking Algorithm

```
Let $K_s$ be the crowd set for the current interval and $K_{s-1}$ the previous.
for Every crowd $C_{si} \in K_s$ do
    if $C_{si}$ is a newly discovered crowd then
        Create a green node in $G_c$.
    else
        Get the parent crowd $C_{s-1j} \in K_{s-1}$ with maximum common users with
        $C_{si}$. If there is more than one crowd with same number of common users,
        select the older crowd.
        Create a new blue node and add a directed edge from the parent crowd
        in $K_{s-1}$ to $C_{si}$.
    end if
end for
for Every crowd $C_{s-1j} \in K_{s-1}$ that does not have a child node do
    Change the color of $C_{s-1j}$ to red from blue.
end for
```

K_{s-1} as input. For every crowd $C_{si} \in K_s$, it determines the parent crowd in K_{s-1}. It then adds a directed edge from the parent to the child crowd. The pseudocode for this algorithm is given in Algorithm 3.

Examples of how parent crowds are selected is shown in Figure 2.4. (I) shows two crowds merging. Here, the parent of the crowd in t_{s+1} is the crowd in t_s that contributed the maximum nodes to the crowd in t_{s+1}. (II) shows a single crowd in t_s being split into two crowds in t_{s+1}. The parent of the two crowds in t_{s+1} is obtained directly. (III) shows a case where three crowds in t_s contribute to three crowds in t_{s+1}. Though crowd A and B contribute two nodes each to D, B is designated as the parent of D since B is older than A.

2.4 Experiments and Results

In this section we present the results of four sets of experiments: (i) we first explore the impact of locality-based crowd discovery compared to the static graph clustering approach without the locality optimizations; (ii) then we investigate the

Property	**Total**	**Per hour avg.**
Users	711,612	18,713
Total tweets	61,314,203	27,769
Messages (@$< u >$)	20,394,030	9,236
User pairs	3,756,619	9,310

Table 2.1: Twitter dataset properties.

impact of the tunable edge decay parameter; (iii) we then examine the features of the discovered crowds, including size and lifespan; and (iv) we illustrate some crowd-based trends that differ from trends aggregated from individual users.

2.4.1 Twitter Dataset

To study crowd detection in a real-world setting, we focus on the Twitter micro-blogging service. Through a mix of crawling and API calls to the Twitter service, we collected a sample of tweets from October 1^{st} to December 31^{st}, 2008, accounting for 2208 hours (see Table 2.1 for details). The dataset includes over 710,000 users and over 61.3 million status updates ("tweets") of 140 characters or less. Users can annotate their tweets via the inclusion of hashtags (e.g., "#redsox") to indicate a particular topic. Similarly, users can include *@mentions* of the form @$\langle username \rangle$ within a tweet to reference another user. While these *@mentions* can serve many purposes, the most popular use is as a simple messaging framework, so that a message posted by user u_1 including @$\langle u_2 \rangle$ is considered a message from u_1 to u_2.

Of the, 61.3 million tweets in the dataset, 20.4 million contain the @$\langle username \rangle$ syntax and are considered messages from one user to another. 3.7 million pairs of users are connected by these messages. The hourly distribution of tweeting users, user pairs, and messages sent is shown in Figure 2.5. All are strongly correlated, following a clear daily and weekly patterns.

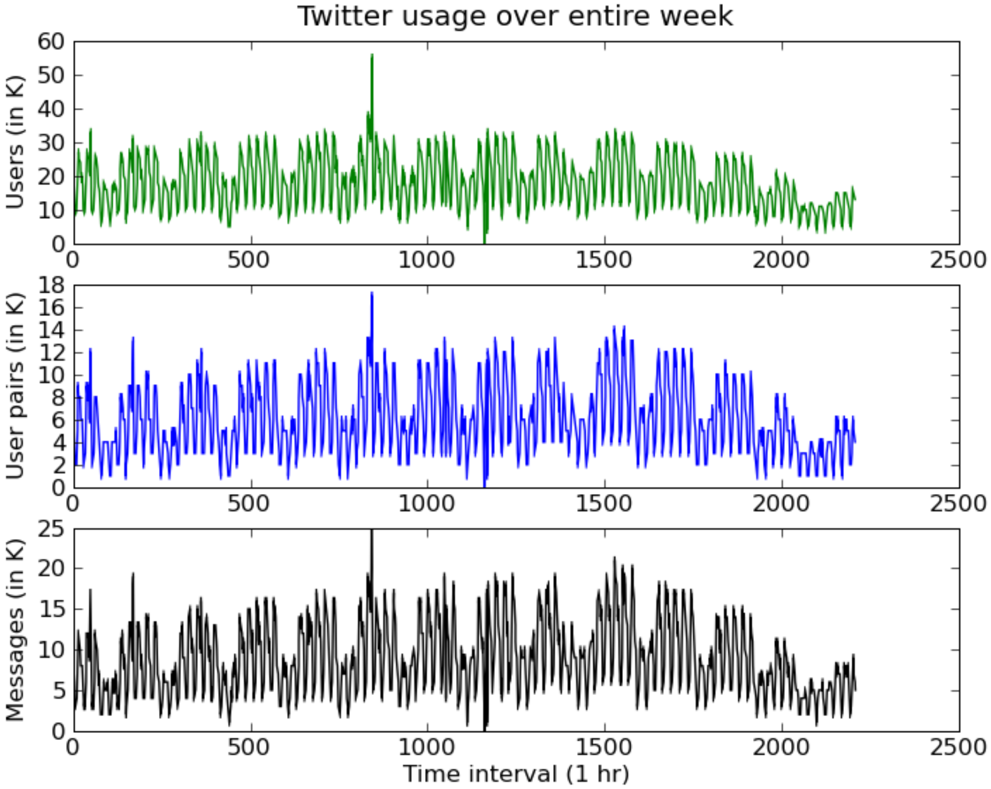


Figure 2.5: Running time comparison.

2.4.2 Performance of Crowd Discovery Algorithm

In the first set of experiments, we investigate the efficiency and quality of the proposed locality-based clustering approach for crowd discovery. Since social messaging systems are large with a high rate of new messages, it is important for crowd discovery to be efficient; but efficiency must be balanced with the quality of the discovered crowds. As a baseline for comparison, we considered the min-cut clustering algorithm [27] without the locality-based optimizations. Since min-cut clustering is designed for static graphs, we took snapshots of the time-evolving communication network every hour and then ran min-cut clustering over each of these hourly

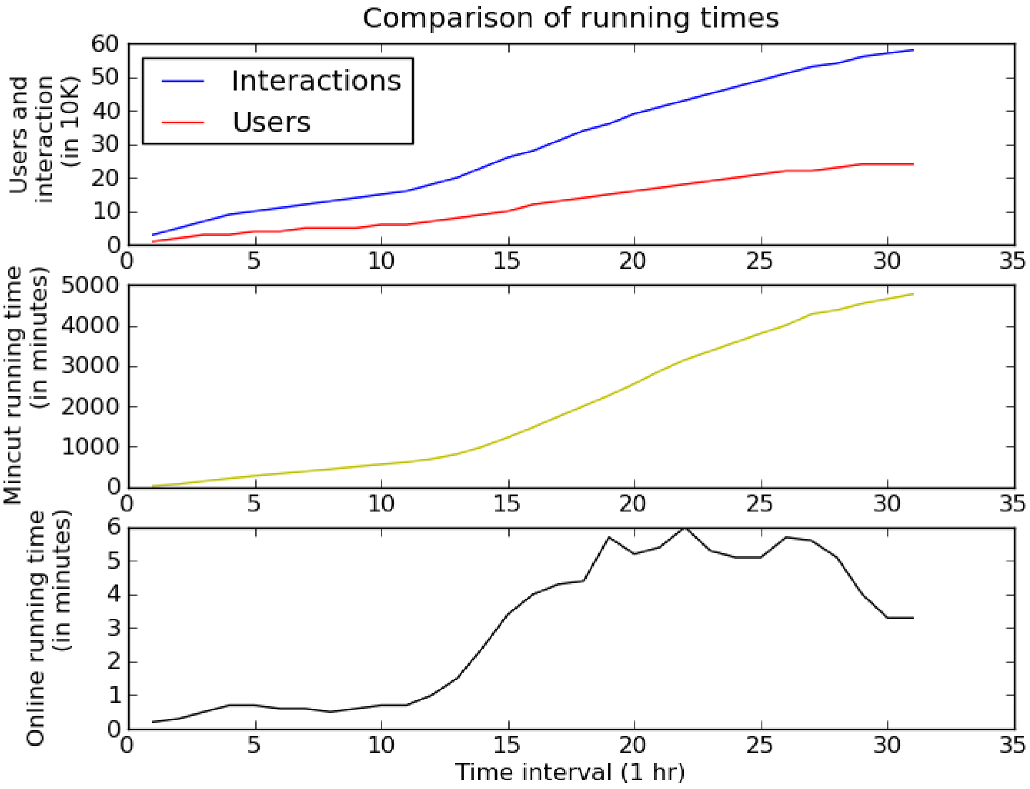


Figure 2.6: Users, user pairs and messages in Oct-Dec, 2008.

snapshots, resulting in 2208 total crowd sets.

Running time: In Figure 2.6, we show the running time comparisons between min-cut clustering and the locality-based crowd discovery approach (note that we focus on the first 30 hours for presentational detail; the general trends hold across the duration). The top plot in Figure 2.6 shows the growth in users and messages; the middle plot shows the running time of min-cut clustering; the bottom plot shows the running time of online clustering algorithm. The first observation is that the proposed approach is at least 100 times faster than non-locality optimized approach in all cases, and upwards of 1,000 times faster in some cases. Next, we observe the impact the

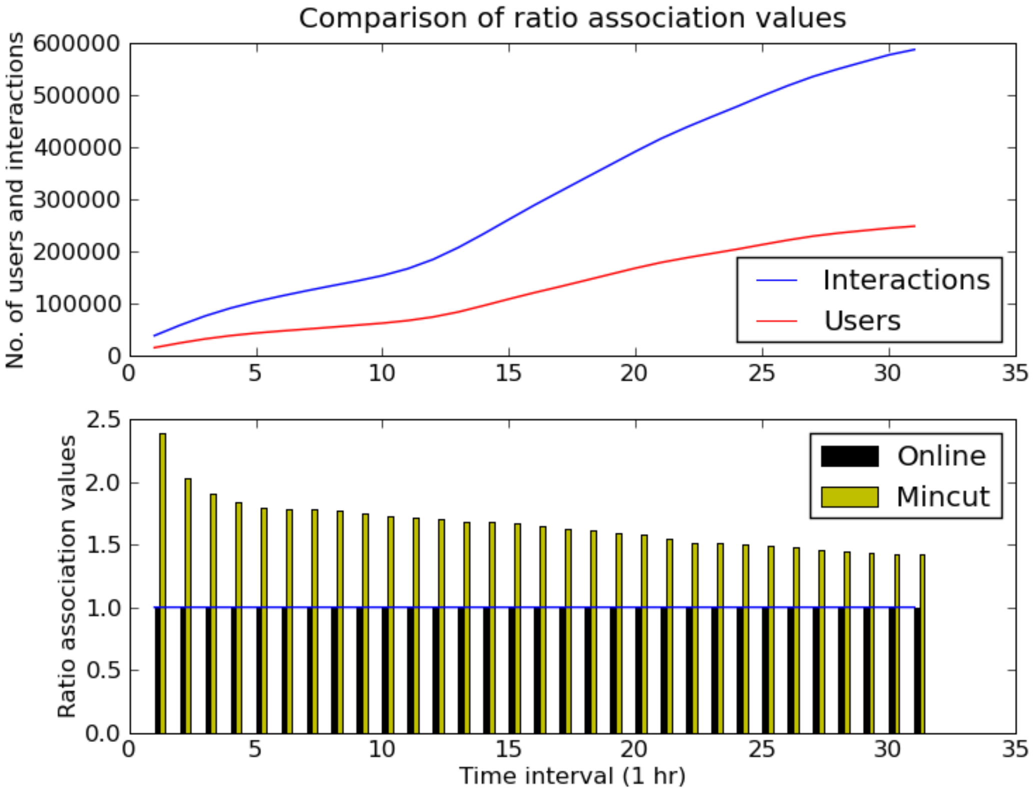


Figure 2.7: Quality comparison using ratio association.

growing number of users and interactions has on the running time of these algorithms. We see that the running time of the min-cut algorithm is proportional to the increase in users and interactions, while our algorithm, because of its locality optimizations, has almost a constant running time. Spatial locality allows our algorithm to cluster a relatively small part of the graph and temporal locality reduces the number of edges by removing old edges.

Crowd quality: Although the proposed locality-based approach results in a much faster crowd discovery, there may be a cost in terms of crowd quality. To gauge this cost, we measure the quality of the discovered crowds using the ratio-association value

[21], which seeks to maximize the weight of edges within a cluster: maximize $\sum_{i=1}^{k} \frac{c(C_i,C_i)}{|C_i|}$. Using this objective, we measure the ratio-association values for both min-cut clustering and the proposed approach. In Figure 2.7, we show the *ratio* of ratio-association values for both algorithms versus the proposed approach; the ratio-association value for local-clustering (online) is indicated using black bars of height 1. We see that during the initial intervals, the ratio-association of the min-cut algorithm is more than that for the locality-based approach, but the ratio continues to decrease with time. We see significant improvements by the time we reach the 30^{th} interval. This shows that as the size of the graph grows the quality of clusters generated by the locality-based approach increases.

Empirically, we find that the locality-based approach supports efficient crowd discovery while maintaining crowds of relatively high quality (within 50% of the ideal case using static graph clustering).

2.4.3 Varying the Edge-weight Decay Coefficient

In the second set of experiments, we analyze the performance of the algorithm as the decay coefficient is modified, from 0.5 to 1.0 to 1.5. The decay coefficient is an important tunable parameter that determines the rate at which crowds disperse. We first show the impact varying this parameter has on the number of crowds discovered and the size of these crowds. We then investigate the impact of this parameter on the speed of crowd discovery and the quality of the crowds discovered.

Impact on number of crowds discovered and crowd sizes: The effect of varying decay co-efficient on crowd size and count is shown in Figure 2.8. We find that the number of crowds discovered for coefficient of 0.5 is more than the ones discovered for 1.0 and 1.5. In the case of larger coefficient values the crowds disperse quickly and hence we find fewer crowds. Coefficients 1.0 and 1.5 discover almost the

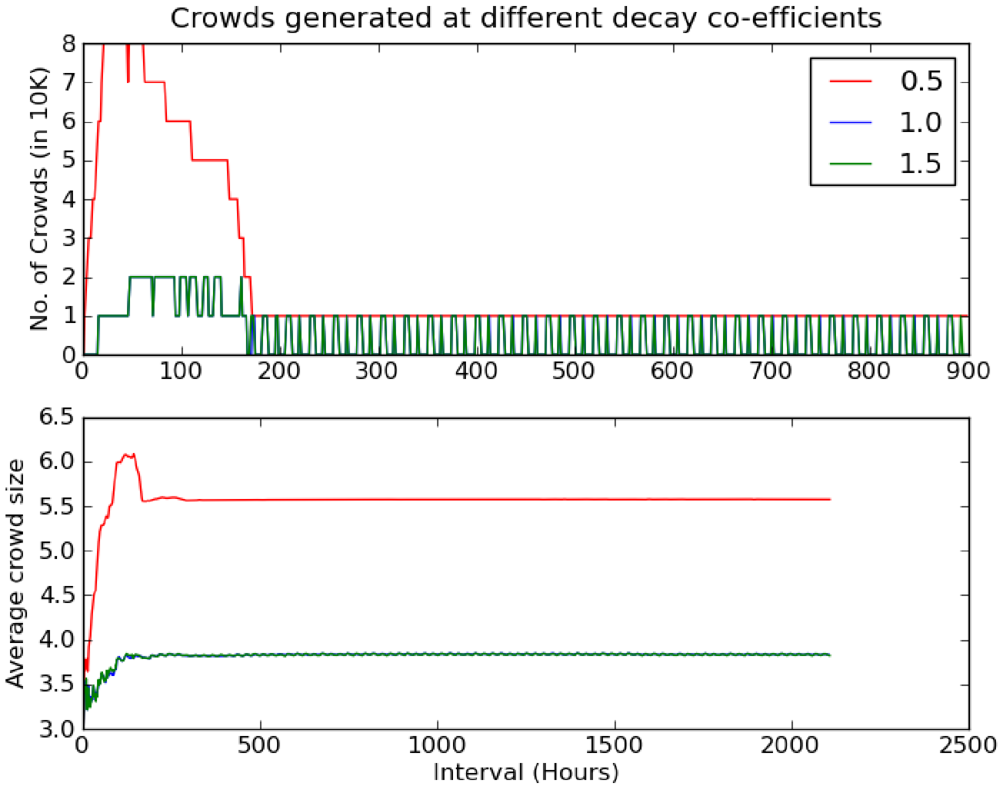


Figure 2.8: Crowd count and size.

same number of crowds. This might be because the crowds that are discovered at 1.0 stay together even at 1.5. It is possible that they disperse at higher co-efficients. We also observe larger crowd sizes at lower coefficient values as the crowds disperse slowly.

Impact on ratio association values: The effect on quality of crowds discovered at different decay coefficient values is shown in Figure 2.9. To observe the quality of crowds discovered we use ratio association, as defined before. We observe that the best crowds are obtained when the decaying coefficient is 1.0. Hence, for the rest of the experiments we set the coefficient to 1.0.

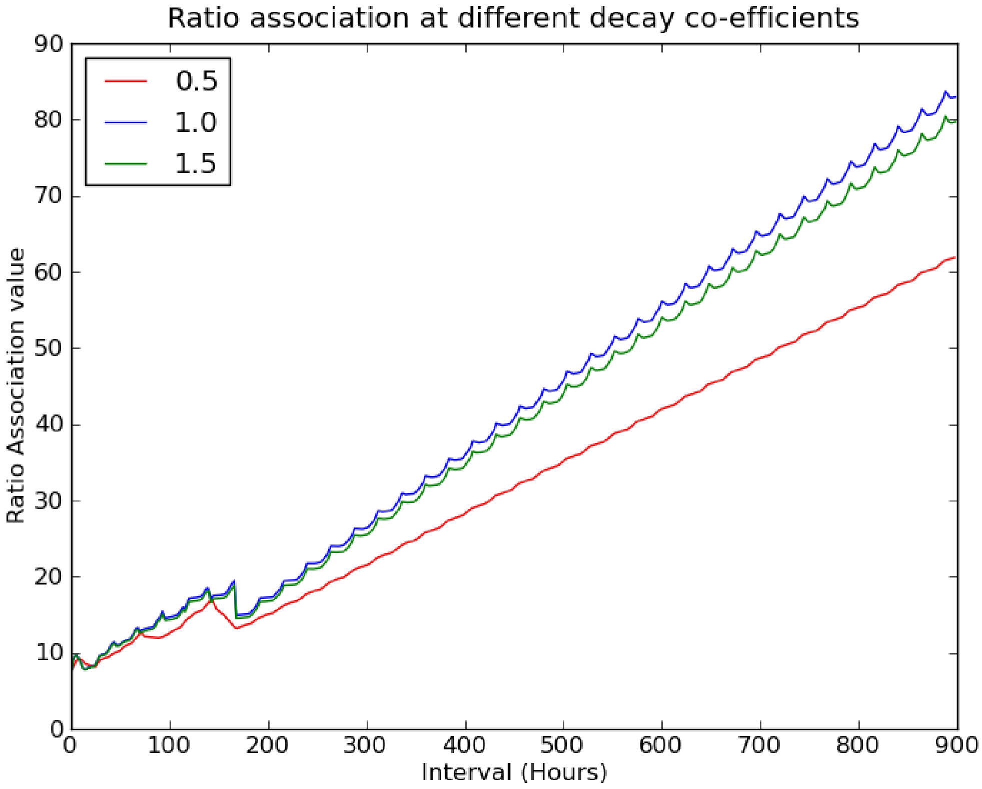


Figure 2.9: Quality of clustering.

Impact on crowd discovery time: We observe that the running time of the algorithm is not dependent on the coefficient (see Figure 2.10). This is an important result because we can now use our algorithm to observe crowds at degrees of granularity by changing the coefficient without affecting the running time performance of the algorithm.

2.4.4 Transient Crowd Analysis

In the third set of experiments, we explore the characteristics of the discovered crowds using the proposed crowd discovery and tracking approach. We identify topics for a particular crowd (akin to the "Crowd Analysis" column in the example

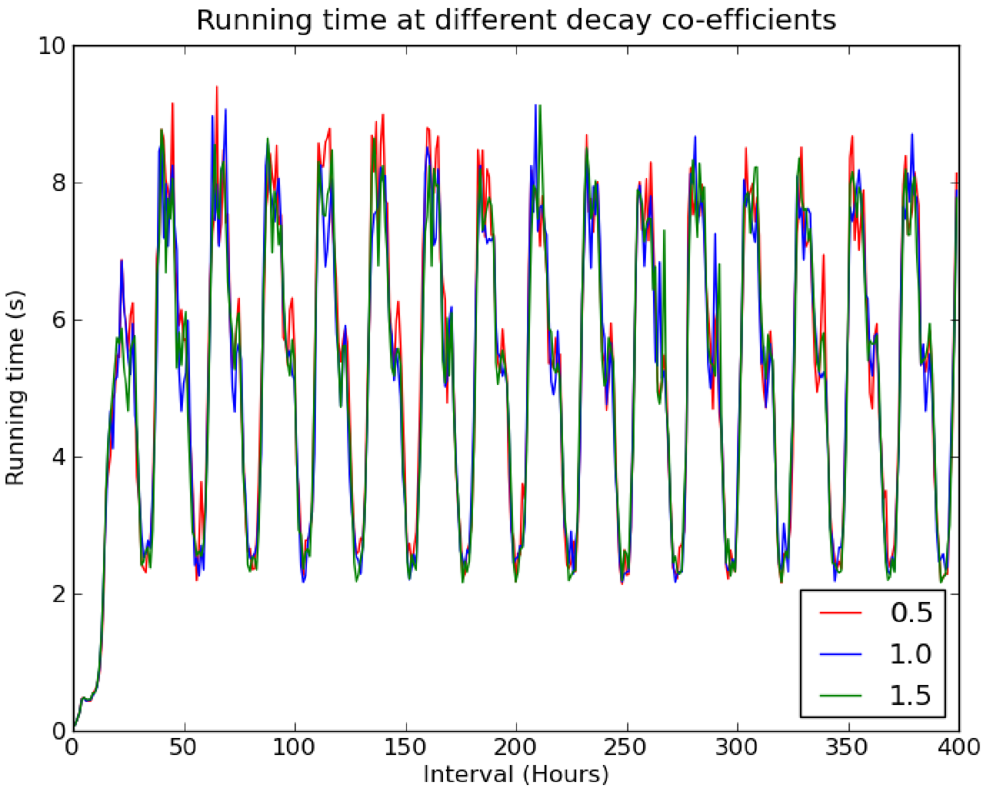


Figure 2.10: Running time of the clustering algorithm.

in Figure 2.1) using a simple approach in which we characterize the topic of a crowd by extracting the nouns from the messages (tweets) exchanged by a crowd.

Time-dependent crowding patterns: We first consider the number of crowds discovered in each time interval. This knowledge can yield insights into crowding patterns in social networks. Figure 2.11 shows the distribution of crowds during a particular week. Like the user and message frequency in Figure 2.5, we observe crowds following a daily pattern. But unlike the previous case, where we saw high and uniform usage throughout afternoon and evening, we observe the largest number of crowds forming in the evening. We are interested to explore this tension between

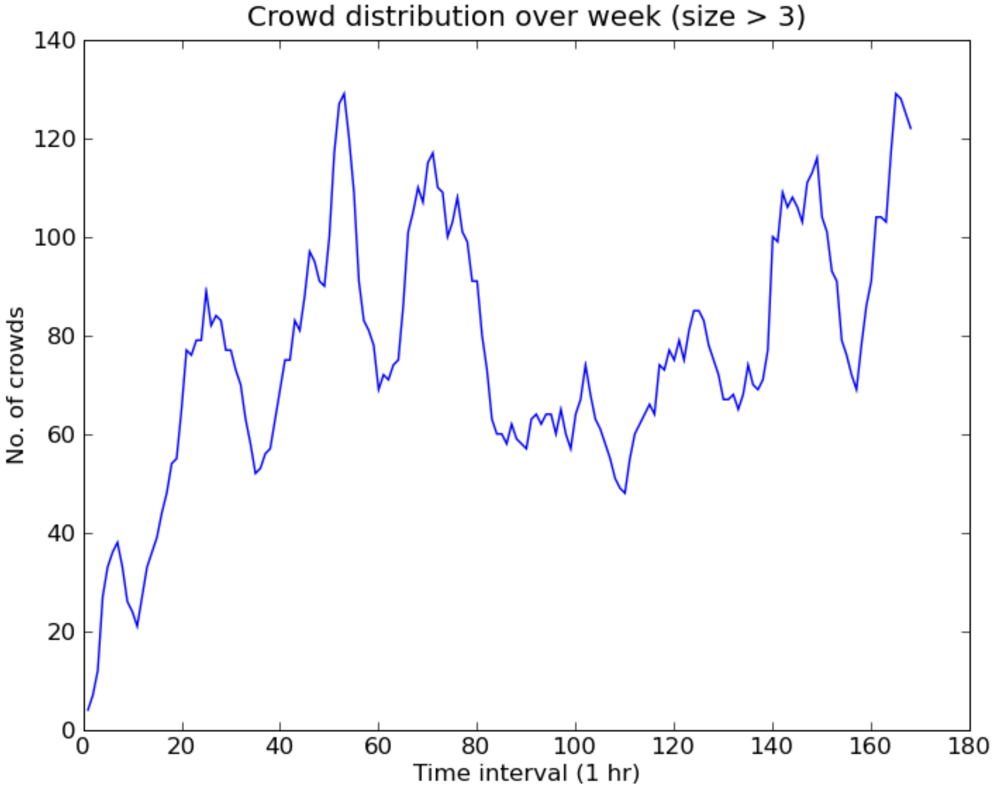


Figure 2.11: Crowds at each time interval.

crowding behavior and overall Twitter usage in our continuing work.

Crowd lifespan: Next, we consider the lifespan of crowds. The lifespan for a crowd can be obtained from the crowd tracking graph discussed in Section 2.3.4. The length for which a crowd lasts is an indicator of its activeness. For example, a crowd that is constantly communicating lives for a longer time than an inactive crowd which disperses. We illustrate some of the discovered crowds and their lifespans in Figure 2.12, with an annotation next to the crowd peak showing the topic of discussion. We see a crowd (shown in black) discussing Sarah Palin and the Vice-Presidential debate from the 40^{th} hour to 80^{th} hour that peaks around the time

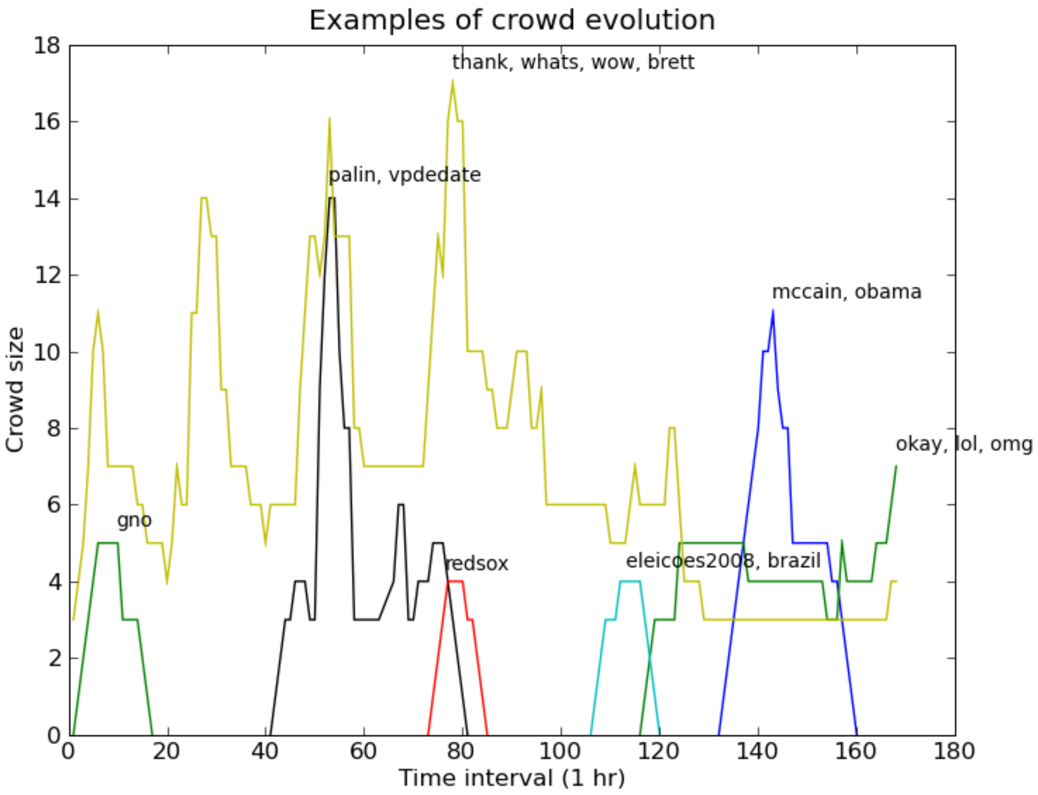


Figure 2.12: Examples of the crowds discovered in the dataset.

of the actual debate. We observe that crowds that talk about general everyday things have a greater lifespan than crowds discussing specific events. For example in Figure 2.12, a crowd (annotated with *thank, whats, wow*) discussing everyday things lives through the entire week, while, during the same period we observe several event-specific crowds, like crowds discussing the Red Sox, Sarah Palin, and Girl's Night Out (gno) forming and dispersing. These event-specific crowds start forming just before the event and die a few intervals after the completion of that event. This distinction between the crowds discovered clearly indicates two types of Twitter usage: first, it is used as a platform to discuss and debate specific events, and second, it as also

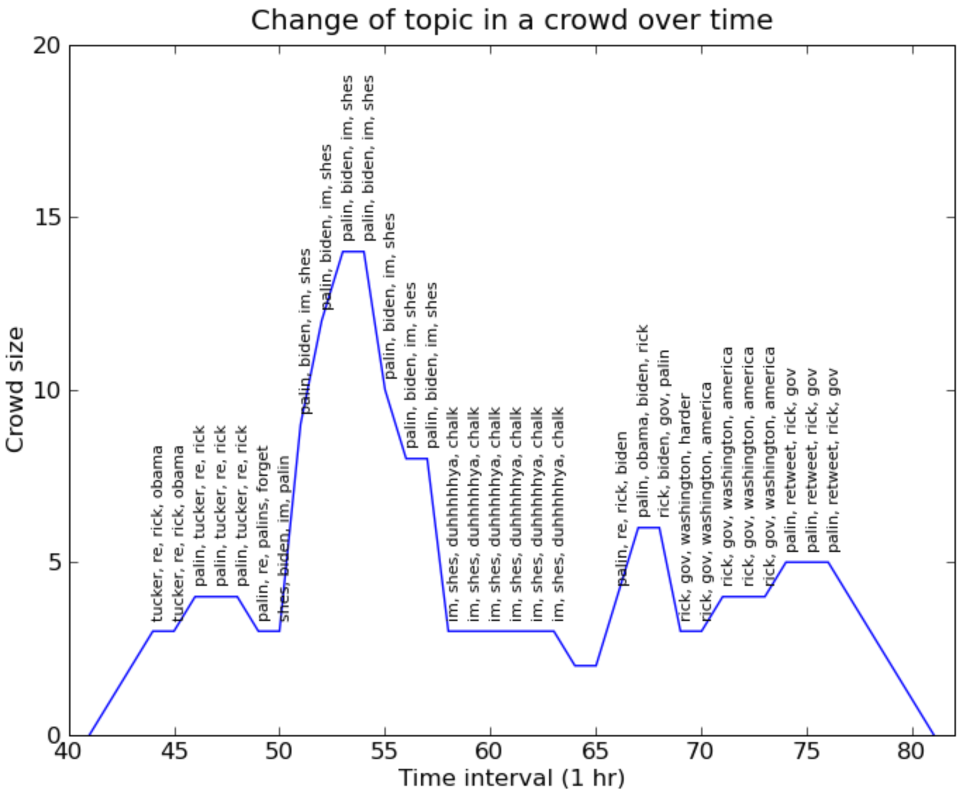


Figure 2.13: Topic evolution in a crowd over time.

used a means of everyday communication.

In the final set of experiments, we compare the topics that interest crowds versus topics that are discovered through the (non-crowd) aggregation of tweets from individual users.

Hashtags vs. Crowd topics: Twitter supports the inclusion of meta-data in tweets through the use of hashtags (e.g., "#redsox"). We first aggregated all of the hashtags in our dataset to see what topics were of most interest. These top hashtags are shown in Table 2.2. Most of the topics determined using hashtags are related to specific events, like debate-related hashtags, conference-related hashtags (*wct08,*

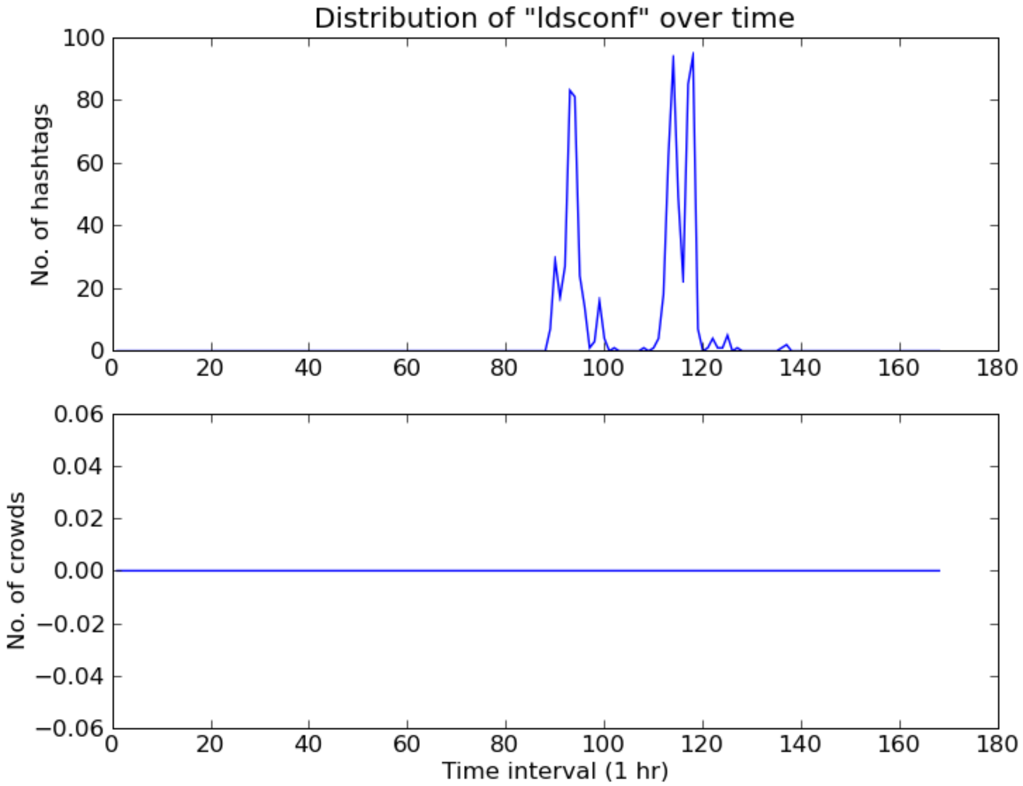


Figure 2.14: Comparison for “ldsconf”

ldsconf, wjs08) etc. This individual-based aggregation is similar to how Twitter’s trending topics works (see http://search.twitter.com/).

In Table 2.2, we also show the topics discovered from our simple noun-based crowd analysis. We see that the crowd-based topics are more varied and less event-specific, like *money*, *kids*, and *school*. Some topics like *ldsconf* (corresponding to the LDS Semi-annual General Conference) are hashtagged often but are part of no crowds (See Figure 2.14). Similar results hold for the conference tags *wcto08, wjs08*, indicating lots of individual activity via tweeting about the conference, but little cohesive communication among members of a community. Another example of the

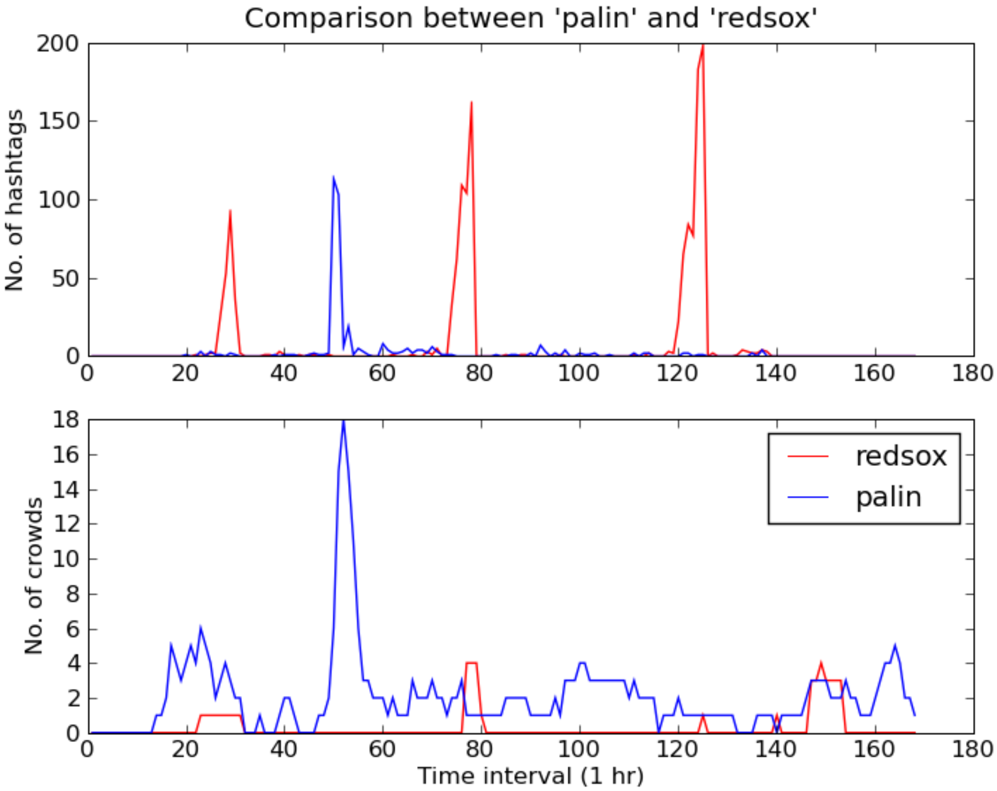


Figure 2.15: Comparison between "redsox" and "palin"

difference between hashtags and topics discussed is shown in Figure 2.15. We see the distribution of the topics *palin* and *redsox*, where the number of hashtags for *redsox* is significantly more than the hashtags for *palin*, but we see that more crowds discuss *palin* than *redsox*.

Topic evolution: Finally, we track the evolution of topics within a crowd as users join and leave over time. Observing the changing topics in a crowd can give us a better understanding about the interests of a crowd and hence help us model the crowd better. An example of such a topic evolution, in a crowd discussing the vice-presidential debate, is shown in Figure 2.13. The crowd at the beginning discusses

Rank	**Hashtags**	**Crowd topics**
1	vpdebate	twitter
2	current	debate
3	redsox	palin
4	vmb	money
5	ldsconf	video
6	debate08	kids
7	palin	obama
8	wcto08	school
9	wjs08	mccain
10	eleicos	office

Table 2.2: Top hashtags and topics observed for the week.

something generic and then starts discussing the Vice-Presidential debate as it occurs (intervals 50-54). The crowd has maximum users during the actual debate and begins to lose users on completion of the debate. As we move away from the debate we see the crowd discussing other topics before dispersing.

2.5 Summary

In this section, we studied the problem of automatically discovering and tracking transient crowds in highly-dynamic social messaging systems like Twitter. We presented a locality-based clustering algorithm for a time-evolving communication network that uses two characteristics of transient crowds – temporal and spatial locality – to support efficient crowd detection. We showed how crowds at different granularity can be discovered by changing edge decay coefficient. We then analyzed these crowds to discover crowd-based topics of discussion, which are different from those identified using hashtags. Finally, with an example we showed how we can track topic evolution in a crowd.

3. DISCOVERY OF CONTENT BASED SOCIAL TRAILS*

In this section, we describe another approach to discover social trails. In Section 2, we described how transient crowds could be viewed through several overlapping perspectives like communication-based, location-based, interest-based, and so on. In this section, we focus on discovery of crowds based on user's interest or the content that they are discussing on social networks.

We propose and evaluate a novel content-driven crowd discovery algorithm that can efficiently identify newly-formed communities of users from the real-time web. Short-lived crowds reflect the real-time interests of their constituents and provide a foundation for user-focused web monitoring. Three of the salient features of the algorithm are its: (i) prefix-tree based locality-sensitive hashing approach for discovering crowds from high-volume rapidly-evolving social media; (ii) efficient user profile updating for incorporating new user activities and fading older ones; and (iii) key dimension identification, so that crowd detection can be focused on the most active portions of the real-time web. Through extensive experimental study, we find significantly more efficient crowd discovery as compared to both a k-means clustering-based approach and a MapReduce-based implementation, while maintaining high-quality crowds as compared to an offline approach. Additionally, we find that expert crowds tend to be "stickier" and last longer in comparison to crowds of typical users.

3.1 Introduction

Long-lived interest based communities, like those on Facebook, Orkut, etc., have been one of the key organizing principles of the Web. The real-time web on the other

*Part of this section is reprinted with permission from "Content-Based Crowd Retrieval on the Real-Time Web" by Krishna Y. Kamath and James Caverlee, 2012. Conference on Information and Knowledge Management. 21st. Copyright 2013 by Association for Computing Machinery (ACM).

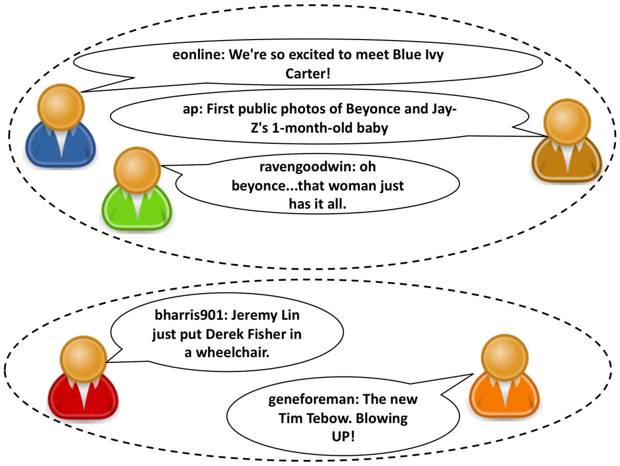

eonline: We're so excited to meet Blue Ivy Carter!
ap: First public photos of Beyonce and Jay-Z's 1-month-old baby
ravengoodwin: oh beyonce...that woman just has it all.
bharris901: Jeremy Lin just put Derek Fisher in a wheelchair.
geneforeman: The new Tim Tebow. Blowing UP!

Figure 3.1: Examples of content based crowds.

hand supports the near instantaneous formation of ad-hoc communities linked by the real-time interests of their constituents. These communities or "crowds" range from groups of loosely-connected Twitter users responding to a live presidential address, to users sharing pictures about a chemical fire at a nearby refinery, and so on. For example, Figure 3.1 shows example of two content based crowds, one discussing the public release of Jay-Z and Beyonce's baby pictures with 3 users (eonline, ap, ravengoodwin), and another crowd about NY Knicks vs LA Lakers basket ball game with 2 users (bharris901, geneforeman).

We first formalize the problem of crowd discovery over rapidly evolving social media and then provide solutions for efficiently identifying crowds. Although we focus on text-based social media streams popularized by Twitter and related services, the discussion and techniques are designed for generic application to other temporally ordered social media resources. Concretely, this section makes the following contributions:

- We present an efficient algorithm for identifying clusters of related users (*crowds*) from the real-time web using a prefix-tree based locality hashing approach.

- We describe an efficient method for updating user profiles in rapidly evolving social media as users post new messages.

- We show how to focus crowd detection via key dimension identification, so that crowd detection can be focused on the most active portions of the real-time web and so resources are not wasted.

- We evaluate the performance of the proposed crowd discovery algorithm over two Twitter datasets and we find the proposed approach is significantly faster than alternative approaches while maintaining high crowd quality.

3.2 Related Work

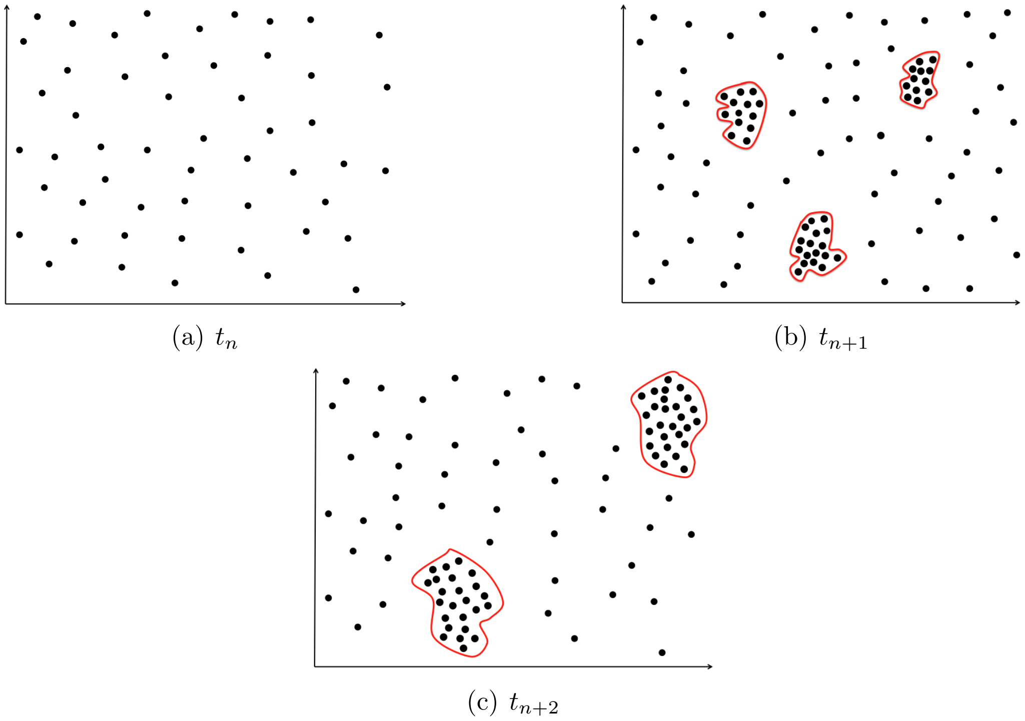
In addition to the works cited in the introduction, there have been many efforts aimed at detecting cluster structure in text-based collections [51, 18, 8]. But, these approaches, however, are typically not designed for high-volume incrementally updated domains as on the real-time web. Alternatively, there is a large body of stream-oriented clustering work for finding correlations in streaming data. For example, StatStream [80] clusters evolving time series data using the Discrete Fourier Transform. Both [2] and [26] explore two-stage approaches for finding clusters in low dimensional data (unlike the case of text clustering, which typically is very high-dimensional due to the number of tokens observed). Clustering over text streams has been studied in [1, 49, 34]. These efforts have focused on the clustering of independent text elements (e.g., new messages), whereas our focus is on finding groups of related users by their sequences of related posts to the real-time web.

The solution approach in this section relies on locality-sensitive hashing (LSH) for finding nearest-neighbors as a primitive for crowd detection. Nearest-neighbor and approximate nearest-neighbor search in a high-dimensional vector space is a difficult problem that Indyk and Motwani [38, 29] approach through the use of a family of randomized hash functions that generate similar vector signatures if the vectors are closer to each other in the high-dimensional space. In [11], Charikar constructed the LSH function for cosine similarity, which supports fast similarity between two high-dimensional vectors by reducing them to bit-arrays of much smaller dimensions. This result has been used in several problems, including efficient noun clustering [63, 53, 59]. In Section 2, we studied crowd detection based on user communication, without regard for the content of the messages as we do here [40].

3.3 Crowd Discovery: Overview and Solution Approach

Let $U = \{u_1, u_2, \ \ldots, u_i \ldots\}$ be a (potentially) unbounded set of users posting messages to a real-time web stream such as Twitter or Facebook. Each user may contribute an arbitrary number of messages, where the messages are ordered in a non-decreasing fashion using the time-stamp values of the messages. We say that a crowd $C = \{u_i, u_2 \ldots u_l\}$, at a given time, is defined as a subset of users that are close to each other at that time, where closeness is measured using a similarity function $sim(u_i, u_j)$. For example Figure 3.2, shows a simple scenario where users are mapped into a 2-dimensional space (say, by using TF-IDF weights of the words in the messages). In the initial figure at time t_n, users are sparsely distributed in the space and there are no clear crowds. As users generate more messages, we see in the following two intervals the formation of several tight clusters of users (“crowds”). Intuitively, these crowds correspond to collections of users who are posting messages about similar topics (e.g., the Super Bowl on one day and Presidential elections the next day).

Given a user similarity measure $sim(\vec{u_i}, \vec{u_k})$ and a user similarity threshold ϵ, we


(a) t_n

(b) t_{n+1}

(c) t_{n+2}

Figure 3.2: Example of user vectors in 2-dimensional space showing the evolution of users during three time intervals. Crowds are shown using a red boundary.

formulate crowd detection as an operation that preserves the following two properties:

Property 1: Every user in a crowd has at least one other user in the same crowd, such that the similarity between them is at least ϵ. That is, $\forall\ u_i \in C \ \ \exists\ u_k : u_k \in C, u_i \neq u_k\ and\ sim(\vec{u_i}, \vec{u_k}) \geq \epsilon$

Property 2: Every user in a crowd has no other user outside the crowd, such that the similarity between them is at least ϵ. $\forall\ u_i \in C\ \neg\exists\ u_k : u_k \in S \backslash C\ and\ sim(\vec{u_i}, \vec{u_k}) \geq \epsilon$

These two properties ensure that (i) all users within a crowd are more similar to users within the crowd than outside of the crowd; and that (ii) there does not exist any user outside of a crowd who is similar to users within a crowd.

By viewing crowd detection in this way, we can avoid memory-intensive ap-

Algorithm 4 Crowd Discovery

```
for $(u, d, t) \in I$ do
    Determine the user nearest to $u$, $u_n$, and the crowd $u_n$ belongs to $C_n$.
    if $sim(\vec{u}, \vec{u_n}) \geq \epsilon$ then
        if $u$ is not in crowd $C_n$ then
            Add $u$ to $C_n$
        end if
    else
        Create a new crowd $C$ with a single user $u$ and add it to $K_t$.
    end if
end for
```

proaches that require maintaining the overall cluster structure (which may be unreasonable for high-volume text); instead, we can formulate the crowd detection problem using nearest-neighbor search as a primitive, as illustrated in Algorithm 4. That is, for every new message posted to the real-time web, we determine the user nearest to the user posting the new message. If the similarity between the user posting the new message and the nearest user is at least ϵ, we add the user to the crowd to which this nearest user belongs, if he is not already in it. If the similarity does not exceed ϵ, we create a new crowd for the given user. K_t is the set of all current crowds at time t. While such an approach may allow long chains of users (where the first user in a crowd is quite distant from the last user), it has the compelling advantage of efficiency.

Towards efficiently discovering crowds from the real-time web, we make note of the following three challenges:

- **Efficient User Profile Updating**: Compared to traditional document clustering, in which documents themselves are static and the goal is to find clusters of related documents, crowd discovery seeks to find clusters of similar users in which users are constantly changing (by posting new messages, changing areas of interest, and so on). Hence, the first challenge is to develop an appropriate representation for users that reflects their current interests accurately and can be easily updated every time they generate a new message.

- **Efficient Crowd Assignment**: The second challenge is to determine an efficient method to determine nearest neighbor for crowd assignment. To find nearest neighbors there are several possible methods (including linear search) and several space partitioning data structures (e.g., k-d trees). However, due to the scale of real-time web updates, such methods may incur a high overhead. Hence, we propose a prefix-tree based locality sensitive hashing method that supports $O(1)$ lookup of a user's nearest neighbor, leading to efficient crowd assignment.

- **Identifying Key Dimensions**: Even with a reasonable method for updating user profiles and assigning users to crowds, the real-time web is constantly growing due to the insertion of new phrases, hashtags, and other artifacts of user-contributed content. Figure 3.3 shows the number of unique tokens encountered over two 10-day Twitter samples (described later more fully in Section 3.4.1 of this section), leading to a linear growth in the dimensions for representing users. Hence, the third challenge is to develop a method to identify important dimensions, so that crowd detection can be focused on the most active portions of the real-time web and so resources are not wasted.

In the following, we approach each of these three challenges in turn, before turning to an experimental evaluation.

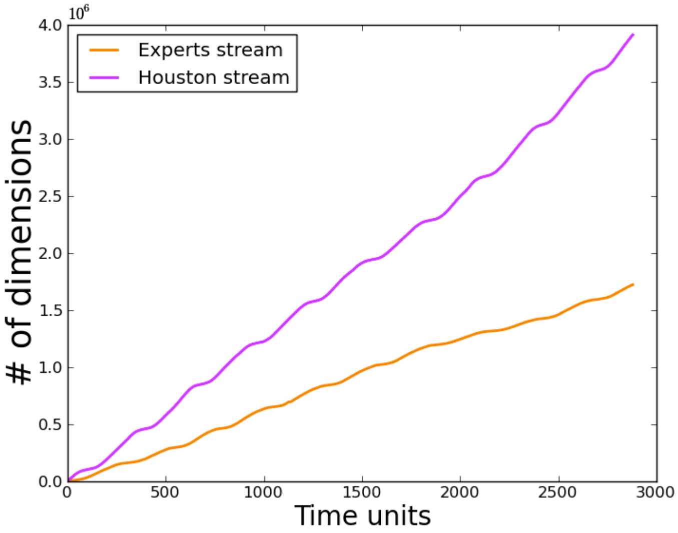


Figure 3.3: Linear growth of dimensions

3.3.1 Efficient User Profile Updating

In this section, we first develop a vector representation for users that decays temporally, so that users are assigned to crowds that reflect their current interests and then we show how to efficiently update these user profiles as new messages are generated.

3.3.1.1 Vector Representation with Fading Memories

Adopting a vector space model for users, let $\vec{u_i}$ be the vector representation for user u_i, where the elements of the vector correspond to tokens parsed from u_i's messages. There are many domain-dependent choices for parsing messages, including language-dependent parsers, entity extraction, stemming, and so forth; for simplicity, we adopt a simple unigram parser that treats all strings separated by whitespace as valid tokens. Since the number of unique tokens corresponding to dimensions are not known in advance, we represent each user profile vector using an infinite co-ordinate space F^{∞} [76]. Under this model, a user u_i at time t is represented as:

$$\vec{u_i}^t = (V_{i1}^t, V_{i2}^t, \ldots, V_{im}^t, \ldots)$$

where the User Vector Dimension (UVD) value V_{im}^t, is the value for u_i in the m^{th} dimension at time t. Let x_{im}^t be the number of times u_i generates m at time t, and $X_{im}^{t_l} = \{x_{im}^1, x_{im}^2 \ldots x_{im}^{t_l}\}$ be the set of all occurrences of m generated by u_i until t_l, then $V_{im}^{t_l}$ is defined as:

$$V_{im}^{t_l} = \sum_{x_{im}^t \in X_{im}^{t_l}} \mathcal{F}(x_{im}, t, t_l) = \sum_{x_{im}^t \in X_{im}^{t_l}} x_{im}^t \tag{3.1}$$

where $\mathcal{F}$ is a function of x_{im}, t and t_l and is called the UVD function.

In this way, a user is represented as the sum of his entire message history. However, since crowds are designed to reflect users with a similar current interest, such an approach may favor crowds of users who are similar in the long-term. For example, we may identify crowds of students, of entertainers, and of politicians, but miss cross-cutting crowds that are drawn together by their current situation (e.g., emergency-oriented crowds reacting to a local earthquake). An alternate approach is to construct user profile vectors using the latest messages only. While such an approach has the advantage of being memory-less (and so, old messages may be dropped with no penalty), grouping users based only on their most recent messages may result in high crowd fluctuation since crowd assignments may vary with each new message.

To balance these two extremes, we propose to adopt a representation that fades user vectors such that recently used dimensions have higher values and older dimensions have lower values. To decay user vectors, we design another UVD function $\mathcal{D}$, which decreases the score of inactive dimensions and increases the score of active dimensions in user vectors. The function $\mathcal{D}$, re-calculates scores for $x_{im}^{t_o}$ at time t_n, as shown:

$$\mathcal{D}(x_{im}, t_o, t_n) = \lambda_u^{t_n - t_o} x_{im}^{t_o} \tag{3.2}$$

where $\lambda_u \in [0, 1]$ is a constant know as the *user dimension score decay rate.* Hence, we can re-write $V_{im}^{t_l}$ as:

$$V_{im}^{t_l} = \sum_{x_{im}^{t} \in X_{im}^{t_l}} \mathcal{D}(x_{im}, t, t_l) = \sum_{x_{im}^{t} \in X_{im}^{t_l}} \lambda_u^{t_l - t} x_{im}^{t} \tag{3.3}$$

Note that when $\lambda_u = 1$, the value of $V_{im}^{t_l}$ is same as that calculated using $\mathcal{F}$ as the UVD function.

3.3.1.2 Efficient Updates

To calculate $V_{im}^{t_l}$ using (3.3), we have to maintain the entire set $X_{im}^{t_l}$. In the context of the real-time web, this can be inefficient since it requires maintaining $X_{im}^{t_l}$ for all users and all dimensions and since the calculation of $V_{im}^{t_l}$ would be O($|X_{im}^{t_l}|$). To solve this problem we prove a proposition that will help us calculate the value of $V_{im}^{t_l}$ efficiently in $O(1)$ time without requiring us to maintain the set $X_{im}^{t_l}$.

Proposition 3.3.1. *If t_{n-k} is the latest time when u_i generated a message with dimension m until t_n, then the value of the dimension at time t_n, is given by:*

$$V_{im}^{t_n} = \lambda_u^{(t_n - t_{n-k})} V_{im}^{t_{n-k}} + x_{im}^{t_n}$$

where, $V_{im}^{t_{n-k}}$ and $V_{im}^{t_n}$ are the values of dimension m for u_i at time t_{n-k} and t_n

respectively.

Proof. Let $X_{im}^{t_{n-k}}$ be the set all occurrences of dimension m in the messages generated by u_i up to time t_{n-k}. Then, using (3.3) we get:

$$V_{im}^{t_{n-k}} = \sum_{x_{im}^t \in X_{im}^{t_{n-k}}} \lambda_u^{t_{n-k}-t} x_{im}^t \tag{3.4}$$

Using (3.2), $\forall x_{im}^t \in X_{im}^{t_{n-k}}$ we can write:

$$\mathcal{D}(x_{im}, t, t_n) = \lambda_u^{t_n - t} x_{im}^t = \lambda_u^{(t_n - t_{n-k}) + (t_{n-k} - t)} x_{im}^t \tag{3.5}$$

$$\mathcal{D}(x_{im}, t, t_n) = \lambda_u^{(t_n - t_{n-k})} \lambda_u^{(t_{n-k} - t)} x_{im}^t \tag{3.6}$$

where t is the time-stamp of every occurrence of m in messages generated by u_i.

Using (3.3) again, we write,

$$\begin{aligned} V_{im}^{t_n} &= \sum_{x_{im}^t \in X_{im}^{t_n}} \mathcal{D}(x_{im}, t, t_n) \\ &= \sum_{x_{im}^t \in X_{im}^{t_{n-k}}} \mathcal{D}(x_{im}, t, t_n) + \sum_{n'=n-k+1}^{n} \mathcal{D}(x_{im}, t_{n'}, t_n) \end{aligned}$$

Using (3.4) and (3.6) we can now write

$$V_{im}^{t_n} = \lambda_u^{(t_n - t_{n-k})} V_{im}^{t_{n-k}} + \sum_{n'=n-k+1}^{n} \mathcal{D}(x_{im}, t_{n'}, t_n)$$

Since u_i did not generate any messages with dimension m after t_{n-k} until t_n, $\forall\ n' \in [n-k+1\ ..\ n-1]$, we have:

$$\mathcal{D}(x_{im}, t_{n'}, t_n) = \lambda_u^{t_n - t_{n'}} x_{im}^{t_{n'}} = \lambda_u^{t_n - t_{n'}} \times 0 = 0$$

Hence,

$$V_{im}^{t_n} = \lambda_u^{(t_n - t_{n-k})} V_{im}^{t_{n-k}} + \mathcal{D}(x_{im}, t_n, t_n)$$
$$V_{im}^{t_n} = \lambda_u^{(t_n - t_{n-k})} V_{im}^{t_{n-k}} + x_{im}^{t_n}$$

Note that, by definition $x_{im}^{t_n} \neq 0$ if u_i generates a message with m, else it is 0. This proves the proposition. □

In brief, we have described an approach to represent users in high-dimensional vector space that reflects their current interests and we have shown how to update this user profile efficiently upon the arrival of each new user message.

3.3.2 Efficient Crowd Assignment

Given the user profile developed in the previous section, we now turn to the challenge of assigning users to crowds as outlined in Algorithm 4. This is the core step in crowd detection and is, in essence, a nearest-neighbor problem. To find nearest neighbors there are several possible methods. The simplest algorithm to determine nearest neighbor is through linear $O(n)$ search, which is not efficient due to the large number of users on the real-time web. Alternatively, we can use efficient space-partitioning methods like k-d trees, which have a complexity of $O(\log n)$.

Here, we propose a specialized variation of the randomized approach to discover nearest neighbors by using locality sensitive hashing (LSH). In this specialized version, we use an additional prefix tree data structure to support $O(1)$ lookup of a user's nearest neighbor, at a cost of requiring $O(n)$ to look up the user's next nearest neighbor. But by constructing crowd detection as a requiring only user's single nearest neighbor (recall the two properties at the beginning of this section), we can support efficient crowd detection over the real-time web.

3.3.2.1 Similarity using Locality-Sensitive Hashing

We first describe a function to calculate the similarity between two vectors using LSH and then describe how we can use this similarity function to determine nearest neighbors efficiently using a prefix tree. Since users are represented as vectors, we can use a metric like cosine similarity to determine the nearest neighbor. But, as described in [38], determining nearest neighbors using cosine similarity is inefficient in high dimensions. Hence, we calculate the approximate cosine distance between two vectors using the approach proposed by Charikar [11].

In [11], the author proposed using LSH functions generated using random hyperplanes to calculate approximate cosine distance. Consider a set of vectors in the collection R^m. Let $\vec{r}$ be a m-dimensional random vector, such that each dimension in it is drawn from a 1-dimensional gaussian distribution with mean 0 and variance 1. Then the hashing function $h_{\vec{r}}$ corresponding to $\vec{r}$ is:

$$h_{\vec{r}}(\vec{v}) = \begin{cases} 1 & \text{if } \vec{r}.\vec{v} \ \geq \ 0 \\ 0 & \text{Otherwise} \end{cases}$$

Now, if we have a set $R = \{\vec{r_1}, \vec{r_2}, \ldots, r_{\vec{|R|}}\}$ of such m-dimensional random vectors, then for a vector $\vec{v}$, we can generate its signature $\bar{v} = (h_{\vec{r_1}}(\vec{v}), h_{\vec{r_2}}(\vec{v}), \ldots, h_{r_{\vec{|R|}}}(\vec{v}))$. Given two user vectors $\vec{u_i}$ and $\vec{u_j}$, the approximate cosine similarity between them is given as:

$$sim(\vec{u_i}, \vec{u_j}) = cos(\theta(\vec{u_i}, \vec{u_j})) = cos((1 - Pr[\bar{u}_i = \bar{u}_j])\ \pi) \tag{3.7}$$

So, the closer the signatures, the greater is the cosine similarity, and the more dissimilar the signatures, the lesser is their cosine similarity. This equation measures approximate cosine distance, and accuracy of this approximation can be improved by using a longer signature, i.e., a larger R.

3.3.2.2 Nearest Neighbor using Prefix Tree

We now describe the procedure to find the nearest user u_n for a user u, from whom we can determine the nearest crowd C_n. We determine u_n using a set of permutation functions $P = \{\pi_1, \pi_2, \ldots, \pi_{|P|}\}$, where each permutation function is of the form:

$$\pi(x) = (ax + b) mod\ p$$

where, p is a prime number and a, b are chosen randomly.

Let $\mathcal{P}$ be a collection of $|P|$ prefix trees, where every prefix tree corresponds to a permutation function $\pi \in P$.

Now, to add a vector $\vec{v}$ to $\mathcal{P}$, first its signature $\bar{v}$ is determined, and then the signature is inserted into every prefix tree in $\mathcal{P}$ after permuting it using the corresponding permutation function. So for a given vector, $|P|$ permutations of its signature are stored in $\mathcal{P}$. Every time we observe a new user vector it is added to $\mathcal{P}$. Similarly, every time we modify a user vector, we remove its old signature from all the prefix trees in $\mathcal{P}$ and add the new one.

To determine the crowd nearest to $\vec{u}$ in $\mathcal{P}$, we first calculate its signature $\bar{u}$. Then for every prefix tree in $\mathcal{P}$, we permute this signature using the corresponding permutation function and find the nearest signature in the prefix tree, by iterating through the tree one level at a time starting from the root. After doing this step we end up with $|P|$ signatures, of which the crowd corresponding to the signature with smallest Hamming distance is picked as the nearest neighbor of $\vec{u}$. As a result, we see that using a prefix tree in combination with LSH, we can design an efficient algorithm to assign users to crowds.

3.3.3 Identifying Key Dimensions

The final challenge is a consideration of the purpose of the crowd monitoring application in the selection of the key dimensions for representing user vectors. For example, if the crowd detection system is intended for topic-focused crowd detection (e.g., identify all "earthquake" related crowds, find all crowds related to "politics"), then the user vectors could be weighted toward these key dimensions (e.g., as in a scheme for weighting the dimensions corresponding to the tokens "obama", "debate", "republican" as more important dimensions than non-politics dimensions). Potential solutions include pre-seeding the crowd detection system with expert-labeled keywords or in identifying high value terms by their *inverse document frequency (IDF)*, which weights key terms by their relative rarity across all documents.

In this section, we propose to select as key dimensions those that reflect the general consensus of the real-time web. That is, we seek to identify tokens that are globally popular at a particular time for biasing the crowd detection toward these tokens. In this way, crowds are defined both by users who have posted similar messages recently (as described in the previous section) and by reflecting topics of great importance to the overall system.

Concretely, our goal is to select from all dimensions the most m significant dimensions. As the real-time web evolves the list of top-m dimensions can then be updated frequently to remove old dimensions and add new ones. Hence, we require a metric to score the dimensions observed so far. To score the dimensions observed in the stream, we use an approach similar to the one used in scoring dimension score for a user vector in Section 3.3.1.

Let y_d^t be the number of times a dimension d appeared in the stream at time t.

Then the score for $y_d^{t_o}$ at time t_n, $t_n \geq t_o$, is given by a function $\mathcal{E}$, defined as:

$$\mathcal{E}(y_d, t_o, t_n) = \lambda_d^{t_n - t_o} y_d^{t_o} \tag{3.8}$$

where $\lambda_d \in [0, 1]$ is a constant known as the *dimension score decay rate.*

Since a dimension can be observed several times in a stream, the score for a dimension d at time t, W_d^t, is calculated as shown in Proposition 3.3.2

Proposition 3.3.2. *If t_{n-k} is the latest time when dimension d was observed on the stream until t_n, then the dimension score for the dimension at time t_n, is given by:*

$$W_d^{t_n} = \lambda_d^{(t_n - t_{n-k})} W_d^{t_{n-k}} + y_d^{t_n}$$

where, $W_d^{t_{n-k}}$ and $W_d^{t_n}$ are the dimension scores at time t_{n-k} and t_n respectively.

Proof. The proof for this is similar to the proof of Proposition 3.3.1. □

Hence, we can identify dimensions that reflect the consensus of the current activity of the real-time web, so that crowd detection can be focused on the most active portions of the real-time web and so resources are not wasted.

3.3.4 Putting it All Together

Taken together, the high-level crowd discovery algorithm described in Algorithm 4 and the three methods developed – efficient user profile updating, efficient crowd assignment, and identifying key dimensions – give us the crowd discovery algorithm in Algorithm 5.

3.4 Experiments

In this section, we report a series of experiments to study crowd discovery. We evaluate the running time performance of the proposed crowd discovery algorithm

Algorithm 5 Crowd Discovery

```
**Create** $R$: Create the set $R = \{\vec{r_1}, \vec{r_1} \ldots \vec{r_{|R|}}\}$ of random Gaussian vectors such that $|R| << m$.
**Initialize** $\mathcal{P}$: Create the set of permutation functions $P = \{\pi_1, \pi_2, \ldots, \pi_{|P|}\}$, where each permutation function is defined using a prime number $p$ and values $a, b$ chosen randomly. Initialize $\mathcal{P}$ as a collection of $|P|$ prefix trees and assign a unique permutation function from $P$ to every prefix tree in $\mathcal{P}$.
for $(u, d, t) \in I$ do
    **Update** $\vec{u}$: Update the user vector $\vec{u}$ using $(d, t)$ as described in Section 3.3.1. Generate new signature for $\vec{u}$ and add or replace it in $\mathcal{P}$.
    **Generate** $\bar{u}$: Generate the $|R|$-bit signature for $\vec{u}$, $\bar{u}$ using $R$.
    **Step 1: Determine** $u_n$ **and** $C_n$: Get the user nearest to $u$, $u_n$ and the crowd $u_n$ belongs to $C_n \in K_t$.
    if $sim(\vec{u}, \vec{u_n}) \geq \epsilon$ then
        if $u$ is not in crowd $C_n$ then
            **Step 2: Add** $u$ **to** $C_n$: Add $u$ to crowd $C_n$.
        end if
    else
        **Step 3: Create** $C$: Create a new crowd $C$ with a single user $u$ and add it to $K_t$.
    end if
end for
```

with other algorithms for crowd discovery. We define metrics to measure quality of crowds discovered and using these metrics we evaluate the quality of crowds discovered by several crowd discovery algorithms. We study the factors impacting the performance of the proposed algorithm, and finally we analyze the properties of crowds discovered over two Twitter datasets.

3.4.1 Dataset

To simulate a Twitter stream, we selected a set of Twitter users and crawled their tweets using Twitter API. The users in this set are labeled using 4 classes – technology, entertainment, politics and sports. To collect this labeled dataset we used the snowball sampling approach. This approach is as follows:

- First, for every class we selected a set of 5 Twitter users, called seed users, that belong to this class and 5 key words that describe the class. For example, for the class sports, a seed user was "espn" and a keyword was "sports".
- We then used the Twitter API to select all Twitter lists that contain a seed user, such that the list's name contains a class specific key word. For example if "sports_news" and "news" are Twitter lists that contain "espn", then we select "sports_news" but not "news", since the former has the keyword "sports" in its name.
- We then extracted a set of new users from the lists selected in previous step and crawled their lists like before.

Following these steps resulted in a "snowball" or chain of crawling actions, which we stopped once we observed sufficient users. At the end of this crawl, we were left with a set of users and the lists they belong to. Every list is also labeled with the class it belongs to. Using this information, for each domain we selected around 1,200 top users and used their tweets to simulate a labelled Twitter stream, resulting in about 1.6 million tweets for 30 days. A similar approach for sampling class specific Twitter data is described in [77]. In addition to this dataset (which we shall call the Experts dataset), we collected a location-based dataset of users tweeting from the Houston region who were selected through random sampling. A 30-day sampling of this stream had about about 15 million tweets from about 107 thousand users. We use the Experts dataset for all of our experiments, except for the experiments in Section 3.4.7 of this section.

3.4.2 Setup

We compare the crowd discovery algorithm (*CDA*) proposed in this section with four alternatives: k-means clustering (*k-means*), a Map-Reduce implementation of k-means clustering (*MR k-means*), a deterministic batched version of the CDA approach (*Iterative-CDA*) – in which we iterate through all the pairs of user vectors to find the best crowds possible, and a Map-Reduce implementation of Iterative-CDA (*MR-CDA*).

For user vector processing, we set the following parameters: number of dimensions $m = 199,999$, user dimension score decay rate $\lambda_u = 0.75$ and dimension score decay rate $\lambda_d = 0.75$. For efficeint crowd assignment, we set signature length $|R| = 23$, number of permutation functions $|P| = 13$ and $\epsilon = 0.005$.

In initial experiments, we varied the choice of k for k-means, finding in many cases that k-means identified many singleton crowds. For the experiments reported here, we set the number of clusters as $k = 0.95\times$number of items to cluster.

3.4.3 Running Time Analysis

To evaluate the running time performance of the proposed approach, we perform two experiments: (i) we use tweet sets of varying sizes as input to all the algorithms

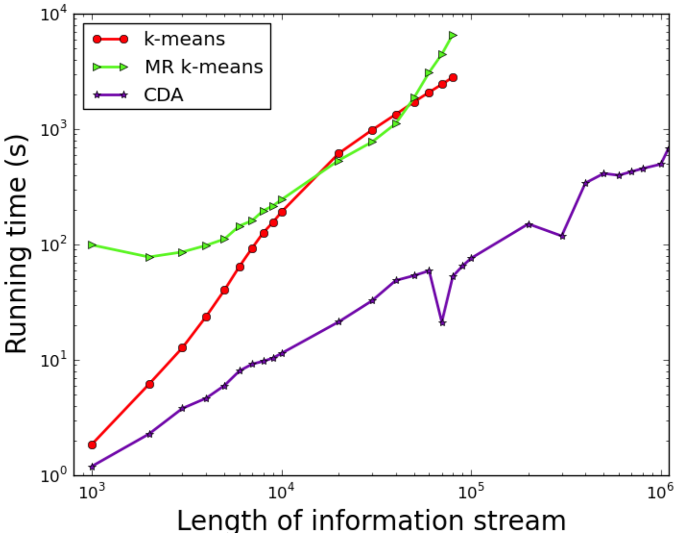


Figure 3.4: Comparison with k-means

and determine the time taken by them to discover crowds; (ii) we measure the tweet processing rate of the algorithms. For these experiments we use a 30 day sample of the Experts stream.

Running Time with Clustering Algorithms: The plot in Figure 3.4, shows the running times for the two k-means clustering algorithms and CDA to discover crowds on data collection of varying sizes. The running times graph is a log-log graph, hence there are orders of magnitude difference between the running times of the algorithms. We see that the time required to discover crowds using the proposed algorithm is significantly lesser than that required by the clustering algorithms. As the size of the message collection increases, both the clustering algorithms become slower. This behavior is expected in case of iterative k-means, because of the extra iterations required by the algorithm, but was not expected in the Map-Reduce version. Generally, the Map-Reduced running time increases at a much slower rate, but is still lesser than that of the iterative version. We believe the worsening performance is because

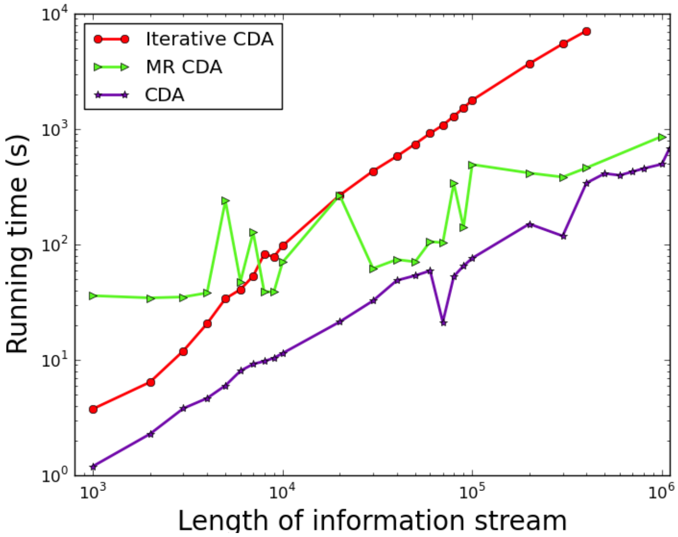


Figure 3.5: Comparison with CDA

of the large value of k. Larger k results is passing of greater number of centroids to a map job which slows down the algorithm. Hence, either of these algorithms are not efficient to discover crowds.

Running Time with CDA Algorithms: We now run similar experiments with the other crowd discovery algorithms. As in the case of the clustering algorithms, we see that CDA, in Figure 3.5, performs much better than the batched CDA algorithms. The Iterative-CDA performs the worst while the MR-CDA performs better after about 10^4 messages. The bad performance of MR-CDA on initial message sets can be attributed the time spent by the MR cluster in setting up the job and passing messages between various workers.

Message Processing Rate with CDA Algorithms: To compare the rate at which the algorithms process messages as they arrive, we note the number of messages that the algorithms have processed at equally spaced time intervals. This

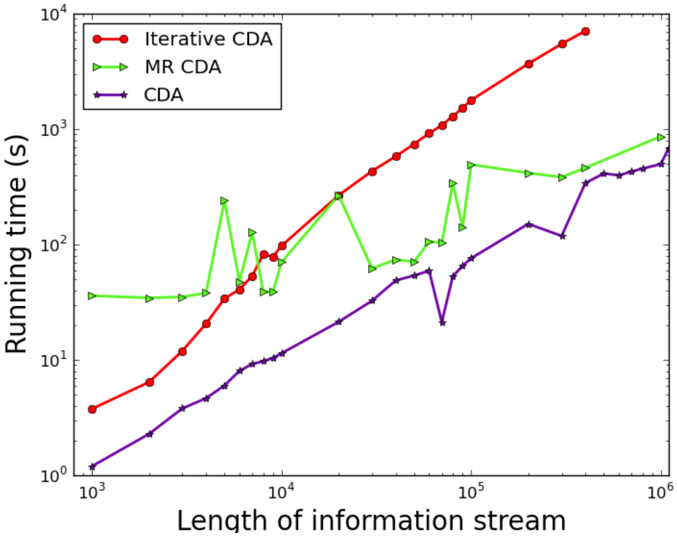


Figure 3.6: Message processing rate comparison of CDAs

comparison is shown in Figure 3.6. As expected, we observe that the number of messages processed by the proposed algorithm is more than that for the other CDA algorithms. This result supports the result we observed with running time Figure 3.5. Similar results were observed for k-means clustering as well but are omitted due to the space constraint.

3.4.4 Crowds Quality Analysis

We now evaluate the quality of crowds discovered using the proposed crowd discovery approach. We know the class to which users in our Twitter stream belong, hence, to evaluate crowd quality we can compare the crowds discovered to this "ground truth". While we do not expect all users belonging to a particular class (e.g., "sports") to form a single large crowd, we do expect that crowds that form will tend to be composed of users belonging to these classes. We use the same 30 day sample of the stream that we used in Section 3.4.3. Like before, the experiments are run with the same value for parameter m. We next describe evaluation metrics that we use to measure quality of crowds and then present performance of CDA against k-means clustering algorithms and deterministic CDAs.

Quality metrics: Consider the set of crowds $K = \{C_1, C_2, \ldots, C_n\}$ for users in set U and a set of classes $\Omega = \{\omega_1, \omega_2, \ldots, \omega_w\}$ to which users in U belong. To measure the quality of crowds generated using crowd discovery algorithms we use the following metrics.

Purity: To compute purity, we assign crowd to the domain which is most frequent in it, and then the accuracy of this assignment is measured by calculating the ratio of correctly assigned users.

$$purity(K, \Omega) = \frac{1}{|U|} \sum_{n} \max_{w} |C_i \cap \omega_j|$$

NMI: Purity gives a good understanding of quality. But, it is susceptible because high purity can be achieved when there are large number of crowds, which we expect in crowd discovery problem. Hence, to deal with this issue, we use a secondary information theory based quality metric called Normalized Mutual Information (NMI). It is defined as:

$$NMI(K, \Omega) = \frac{I(K, \Omega)}{[H(K), H(\Omega)]/2}$$

$$I(K, \Omega) = \sum_{n} \sum_{w} \frac{|C_n \cap \omega_w|}{|U|} \log \frac{|U||C_n \cap \omega_w|}{|C_n||\omega_w|}$$

$$H(K) = -\sum_{n} \frac{|C_n|}{|U|} \log \frac{|C_n|}{|U|}$$

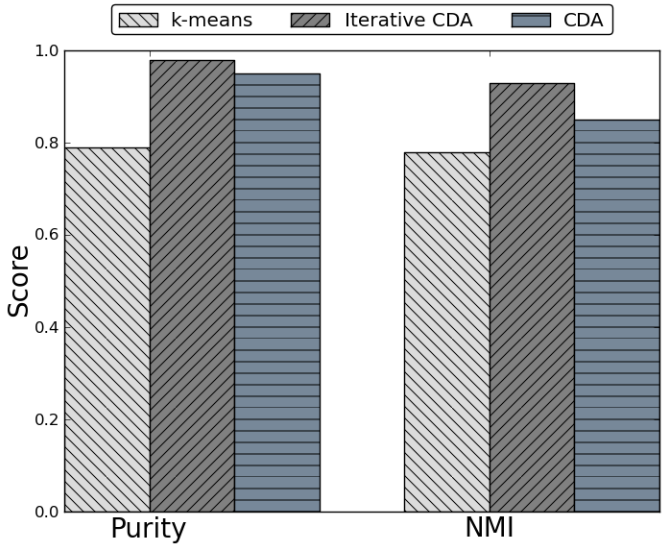


Figure 3.7: Quality of crowd discovery

where, $I(K, \Omega)$ is mutual information and H entropy.

Comparison with Clustering Algorithms: The comparison between quality of crowds discovered using the Iterative k-means and that discovered using CDA is shown Figure 3.7. We see that despite the significant improvements in running time, the crowds discovered by the CDA are still of high quality. We also notice, for all the metrics, the quality of crowds generated using CDA is better than the quality of crowds generated using a clustering algorithm. The relatively poor performance of the clustering algorithm can be attributed to the difficulties in estimating the number of clusters k.

Comparison with CDA Algorithms: The comparison between quality of crowds discovered using the Iterative-CDA and that discovered using CDA is shown in Figure 3.7. We see that crowds discovered by Iterative-CDA are always better than that discovered using CDA. The lower values for these metrics is expected in case of

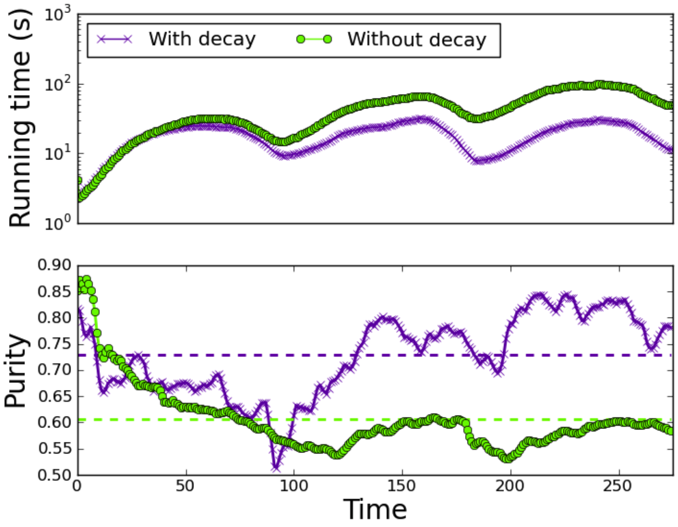


Figure 3.8: User vector representation

CDA, as it is a randomized and an approximate algorithm whereas Iterative-CDA is an deterministic algorithm.

3.4.5 Impact of User Vector Representation

In Section 3.3.1, we described the method to exponentially decay user vectors to help us discover temporally relevant crowds. We evaluate the effectiveness of this approach by analyzing the performance of CDA when the user vectors are exponentially decayed and when they are not. To evaluate the performance of the algorithm without decay, we set $\lambda_u = 1.0$. The difference in quality of the crowds generated by the algorithm using these two approaches is shown in Figure 3.8.

The top plot of Figure 3.8 shows the running time of the algorithms for this experiment. We observe that, thought the running times for the algorithms is almost the same initially, the difference between them increases with time. This is because, as time increases, the algorithm that decays user vector and uses techniques to score

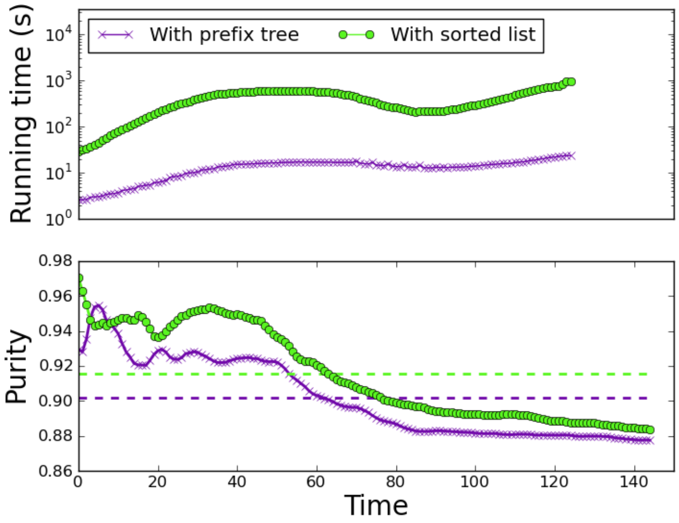


Figure 3.9: Crowd assig. with prefix trees.

dimensions, has the ability to remove dimensions when they become stale. This feature is not possible when the algorithm is run without decay.

As shown in the bottom plot of Figure 3.8, the quality of crowds discovered using exponential decay is much better than the crowds discovered without decay. When user vectors are not decayed, old dimensions are not removed from it, resulting in crowds being discovered which contain users from different domains. This results in lower quality crowds.

3.4.6 Impact of Prefix Trees

We next analyze the impact of using prefix trees on efficient crowd assignment. An alternative approach described in [63] suggests representing $\mathcal{P}$ as a collection of sorted lists of signatures rather than prefix trees. Such a structure is robust in the sense that signatures are sorted and hence nearest neighbor can be found faster than linear search, but has the downside that determining the nearest neighbor and

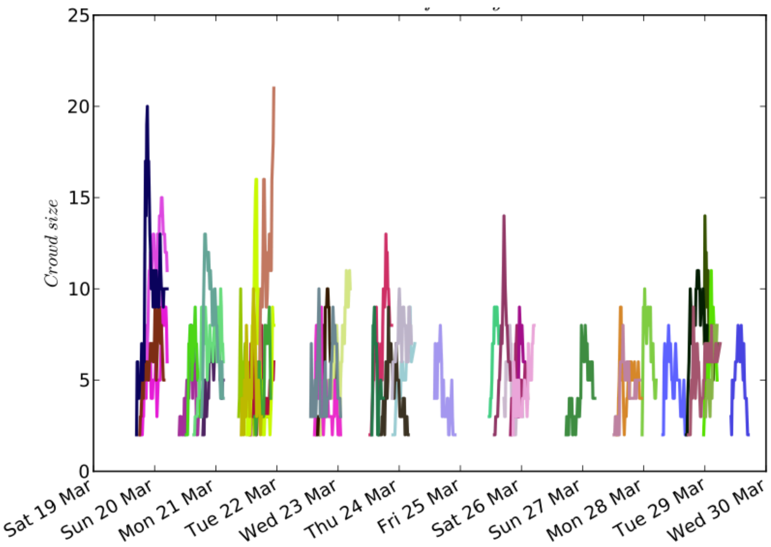
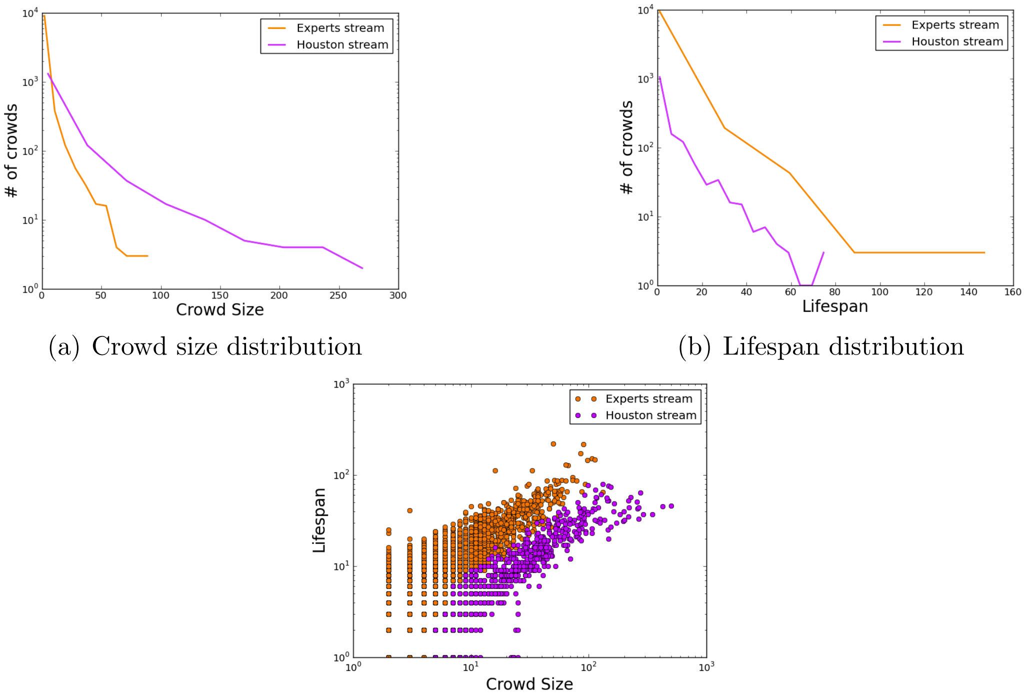


Figure 3.10: Example of crowds related to Libya

adding a new vector takes $O(\log n)$ time, considering $|P|$ is constant. To characterize the impact of the prefix-tree based locality-sensitive hashing approach, we run CDA both with prefix trees and with sorted lists. The results are shown in Figure 3.9.

The top plot shows the running time and the bottom plot shows the quality of crowds discovered. We see that by using prefix trees, we can discover crowds at speeds several times the speed using sorted lists. As mentioned before, the improved speed efficiency is because of the constant time required to retrieve crowds in case of prefix tree instead of $O(\log n)$ as in case of sorted lists.

The quality of crowds generated varies initially when the number of crowds in the prefix tree is small because of randomization involved in determining the nearest neighbor. This variance is overcome as the number of crowds in the prefix tree increases and the mean quality of crowds discovered remains almost the same. After sometime, once we have observed sufficient crowds, we observe that the crowds quality is almost same while using both prefix tree and sorted lists.



(a) Crowd size distribution

(b) Lifespan distribution

(c) Crowd size Vs Lifespan

Figure 3.11: Comparing crowds discovered across the two datasets

3.4.7 Comparing Crowds

Finally, we explore the impact of the kind of users on crowd formation. We compare the crowd size distribution, followed by the lifespan distribution of the crowds. Then we plot these two properties towards understanding crowding behaviors in these two datasets.

The distribution of crowd sizes is shown in Figure 3.11(a). We see that the Houston dataset tends to have larger crowds in comparison to the Experts dataset. These larger crowds may be attributed to the fact that the Houston dataset has relatively more users in comparison to the Experts dataset, and hence more users talking about a particular event resulting in the formation of larger crowds. To understand these dynamics better, we show the lifespan of these crowds in Figure 3.11(b). The lifespan distribution shows that expert crowds, despite being smaller, are mostly longer lasting than the larger crowds discovered in Houston. Based on further analysis, we find that the experts stream is more *sticky* – that is, crowds in the experts stream added new users over time and decayed more slowly.

We attribute this finding to the crowd formation properties of the Experts dataset, whereby crowds are initiated by users who are popular within a particular domain and hence tend to tweet similar things more often. This shared interests among users forms crowds that discuss chains of events resulting in longer lifespans. While users in the Houston dataset form crowds that last only as long as the event they are discussing is popular. This is because Houston has users who have relatively varied interests. Continuing this avenue of investigation, we plot crowd size versus life span in Figure 3.11(c). If the crowds in the Experts dataset are really sticky, as we expect, this should be observed across all the crowds of different sizes, i.e., only larger crowds should not have contributed in making the life span distribution in Figure 3.11(b) appear the way it does. We observe that irrespective of crowd size, expert crowds always seem to have a higher lifespan than Houston crowds. This clearly shows the way users in expert crowds are tweeting and the content of their tweets is making them stick together longer than Houston crowds. In addition to this observation, we also see that the stickiness of the crowds increases with crowd size. This is observed both for the experts and Houston crowds.

We also find that events that last for a long time have more number of crowds that are spread across the event's duration. An example of such a long term event is the revolution in Libya, and crowds related to this appear throughout the experiment duration following a daily pattern based on users activity, as shown in Figure 3.10.

3.5 Summary

In this section, we have seen how the proposed content-driven crowd discovery algorithm can efficiently identify newly-formed communities of users from the real-time web. The approach leverages optimizations to locality-sensitive hashing via prefix trees, incorporates efficient user profile updating, and identifies key dimensions for supporting crowd detection.

4. ANALYSIS OF GEO BASED SOCIAL TRAILS*

In this section, conduct a study of the spatio-temporal dynamics of Twitter hashtags through a sample of 2 billion geo-tagged tweets. In our analysis, we (i) examine the impact of location, time, and distance on the adoption of hashtags, which is important for understanding meme diffusion and information propagation; and (ii) examine the spatial propagation of hashtags through their focus, entropy, and spread. Based on this study, we find that although hashtags are a global phenomenon, the physical distance between locations is a strong constraint on the adoption of hashtags, both in terms of the hashtags shared between locations and in the timing of when these hashtags are adopted. We find both spatial and temporal locality as most hashtags spread over small geographical areas but at high speeds. We also find that hashtags are mostly a local phenomenon with long-tailed life spans. These (and other) findings have important implications for a variety of systems and applications, including targeted advertising, location-based services, social media search, and content delivery networks.

4.1 Introduction

As indicated earlier, the rise of social media services enables a global-scale infrastructure for the sharing of videos, blogs, images, tweets, and other user-generated content. As users consume and share this content, some content may gain traction and become popular resulting in viral videos and popular memes that captivate the attention of huge numbers of users. These phenomena have attracted a considerable

*Part of this section is reprinted with permission from "Spatio-Temporal Dynamics of Online Memes: A Study of Geo-Tagged Tweets" by Krishna Y. Kamath, James Caverlee, Kyumin Lee and Zhiyuan Cheng, 2013. World Wide Web. 22nd. Copyright 2013 by Association for Computing Machinery (ACM).

amount of recent research to study the dynamics of the adoption of social media, e.g., [7, 37, 45, 47, 64].

Augmenting this rich body of research is the widespread adoption of GPS-enabled tagging of social media content via smartphones and social media services, which provides new access to the fine-grained spatio-temporal logs of user activities. For example, the Foursquare location sharing service has enabled 2 billion "check-ins" [28], whereby users can link their presence, notes, and photographs to a particular venue. The mobile image sharing service Instagram allows users to selectively attach their latitude-longitude coordinates to each photograph; similar geo-tagged image sharing services are provided by Flickr and a host of other services. And the popular Twitter service sees ~300 million Tweets per day, of which ~3 million are tagged with latitude-longitude coordinates.

Access to these geo-spatial footprints opens new opportunities to investigate the spatio-temporal dynamics of online memes, which has important implications for a variety of systems and applications, including targeted advertising, location-based services, social media search, and content delivery networks. Hence, in this section, we initiate a study of the *spatio-temporal properties* of social media spread through an examination of the fine-grained sharing of one type of global-scale social media – a sample of 2 billion geo-tagged Tweets with precise latitude-longitude coordinates collected over the course of 18 months. Specifically we consider the *propagation* of hashtags across Twitter, where a hashtag is a simple user-generated annotation prefixed with a #. Hashtags serve many purposes on Twitter, from associating Tweets with particular events (e.g., #ripstevejobs and #fukushima) to sharing memes and conversations (e.g., #bestsportsrivalry and #ifyouknowmeyouknow). Our goal is to explore questions such as:

- What role does distance play in the adoption of hashtags? Does distance between two locations influence both what users in different locations adopt and when they do so?

- While social media is widely reported in terms of viral and global phenomenon, to what degree are hashtags truly a global phenomenon?

- What are the geo-spatial properties of hashtag spread? How do local and global hashtags differ?

- How fast do hashtags peak after being introduced? And what are the geo-spatial factors impacting the timing of this peak?

While limited to one type of social media spread and with an inherent sample bias towards using who are willing to share their precise location, the investigation of these questions can provide new insights toward understanding the spatio-temporal dynamics of the sharing of user-generated content. Our investigation is structured in two steps. First, we study the global footprint of hashtags and explore the spatial constraints on hashtag adoption. In particular, we analyze the worldwide distribution of hashtags and the impact distance has on where and when hashtags will be adopted. Second, we study three spatial properties of hashtag propagation – focus, entropy, and spread – and examine the spatial propagation of hashtags using these properties. Specifically, we study the nature (local or global) of hashtag propagation and the correlation between the spatial properties and the number of occurrences of the hashtags.

Some of our key findings are:

- Hashtags are a global phenomenon, with locations all across the world. But the physical distance between locations is a strong constraint on the adoption of hashtags, both in terms of the hashtags shared between locations and in the timing of when these hashtags are adopted.

- Hashtags are essentially a local phenomenon with long-tailed life spans, but follow a “spray-and-diffuse” pattern [9] where initially a small number of locations “champion” a hashtag, make it popular, and the spread it to other locations. After this initial spread, hashtag popularity drops and only locations that championed it originally continue to post it.

- The rate at which a hashtag becomes popular is dependent on the hashtag’s origin. That is, hashtags that originate as responses to external stimuli (like real-world events) spread faster than hashtags that originate purely within the Twitter network itself (e.g., corresponding to a Twitter meme like #ifyouknowmeyouknow).

These results can positively impact both research into the spread of online memes as well as systems operators, e.g., informing the design of distributed content delivery networks and search infrastructure for real-time Twitter-like content. For example, caching decisions to improve fast delivery of social media content to users and to support applications like real-time search can build upon the results presented here. Insights into the role distance plays and the impact locations have on hashtag spread could inform new algorithms for geo-targeted advertising. This work can also complement efforts to model network structures that support (or impede) the “viralness” of social media, measure the contagion factors that impact how users influence their neighbors, develop models of future social media adoption, and so forth.

4.2 Related Work

Our work presented here builds on two lines of research: studies of Twitter and of Twitter hashtags; and geo-spatial analysis of social media.

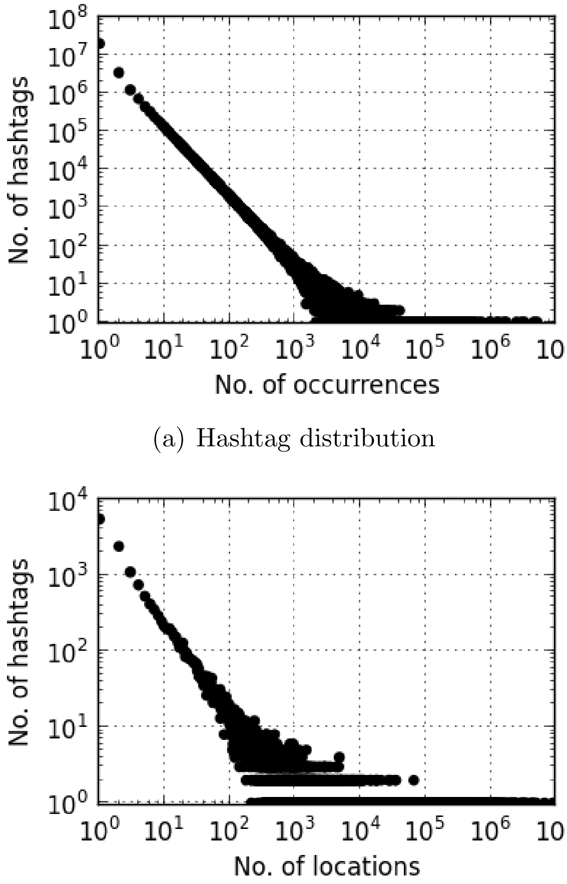
Twitter Hashtag Analysis: There have been several papers studying the general properties of Twitter as a social network and in analyzing information diffusion over this network [37, 45, 78, 46]. Continuing in this direction most papers related to hashtags have focused their attention on understanding the propagation of hashtags on the network. For example, in [64] the authors studied factors for hashtag diffusion and found that repeated exposure to a hashtag increased the chance of it being reposted again, especially if the hashtag is contentious. An approach grounded in linguistic principles has studied the properties of hashtag creation, use, and dissemination in [17]. In related research, approaches based on linear regression have been used to predict the popularity of hashtags in a given time frame [73]. Because of the variety of ways in which hashtags are used to convey information about a tweet, there has been recent research in hashtag-based sentiment detection [20], topic tracking on twitter streams [48], and so forth.

Geo-spatial Analysis of Social Media: The emergence of location-based social networks like Foursquare, Gowalla, and Google Latitude has motivated large-scale geo-spatial analysis [67, 56, 13]. Some of the earliest research related to geo-spatial analysis of web content were based on mining geography specific content for search engines [22]. More recently in [4] the authors analyzed search queries to understand the spatial distribution of queries and understand their geographical centers. On Twitter, geo-spatial analysis has focused on inferring geographic information from tweets like predicting user locations from tweets [12] and spatial modeling to geolocate objects [19]. Similar analysis to infer a user's location on Facebook based on their social network has been studied in [5]. Researchers have observed the highly-local nature of video views based on a sample from YouTube [9].

4.3 Data and Setup

We collected a sample of 2 billion geo-tagged tweets containing 342 million hashtags (27 million unique hashtags) from Twitter using the Twitter Streaming API from February 1, 2011 to October 31, 2012. Each tweet in this sample is tagged with a latitude and longitude indicating the location of the user at the time of the posting, resulting in a tuple of the form $<$ `hashtag`, `time`, `latitude`, `longitude` $>$. The expected long tail distribution for hashtag occurrences is shown in Figure 4.1(a).

To support location-based analysis, we divide the globe into square grids of equal area using Universal Transverse Mercator (UTM), a geographic coordinate system which uses a 2-dimensional Cartesian coordinate system to map locations on the surface of the globe [75]. The issue with using an angular co-ordinate system like latitudes and longitudes is that distance covered by a degree of longitude differs as we move towards the pole. In addition, the distance covered by moving a degree in latitude and longitude is the same only at the equator. Hence, it is hard to break globe into grids using this system. UTM on the other hand gives us a system of grids that closely matches distances in metric system making our analysis easier. While varying the choice of grid size can allow analysis at multiple levels (e.g., from state-sized cells to neighborhood-sized ones), we adopt a middle ground by dividing the globe into squares of 10km by 10km. Some grid cells will naturally be densely populated, others will be sparse. Let this set of distinct locations, each corresponding to a square, be represented by the set L. With these locations, we observe in Figure 4.1(b) that the number of hashtags present in a location follows a long tail distribution (e.g., 10,000 unique hashtags are observed in 10 locations; 100 unique



(a) Hashtag distribution

(b) Location distribution

Figure 4.1: Hashtag dataset properties

hashtags are observed in 100 locations), following the expected population density of equal-sized grid cells.

For the rest of the section, we focus on hashtags with at least 5 occurrences in a location and with at least 50 total occurrences across all locations. Since some hashtags may have begun their Twitter life before the first day of our sample while others may have continued on after the last day, we consider both February 2011 and October 2012 as buffer months. Hence, we capture the full lifecycle of hashtags starting on or after March 1, 2011 and ending by September 30, 2012 which focuses

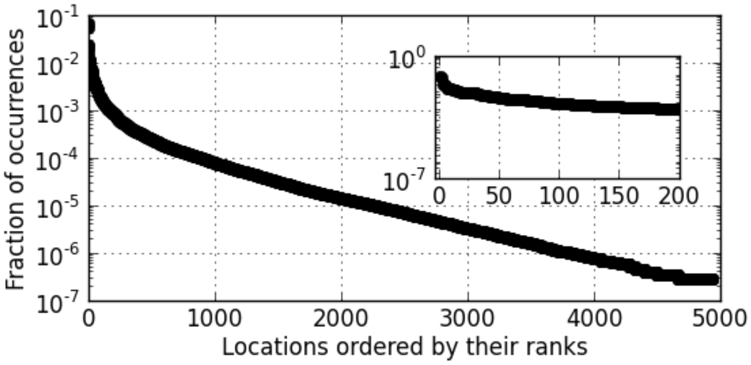


Figure 4.2: Fraction of hashtag occurrences in locations ordered by their rank. The inset plot shows the fraction for top-200 locations.

our study to hashtags that have both their birth and death within the time of study.

The rest of this section considers a set of hashtags H (consisting of close to 20 million hashtags from 99,015 unique hashtags) and a set of locations L (consisting of 4,946 locations). For every hashtag ($h \in H$) and location ($l \in L$) pair, we denote the set of all occurrences of h in l as O_l^h. We say that H_l is the set of unique hashtags observed in l.

Our study continues in three major parts:

- First, we study the global footprint of hashtags and explore the spatial constraints on hashtag adoption. (Section 4.4)
- Second, we study three spatial properties of hashtag propagation – focus, entropy, and spread – and examine the spatial propagation of hashtags using these properties. (Section 4.5)

4.4 Location Properties of Hashtags

In this section, we begin our analysis by examining the locations represented in the dataset and exploring the relationship between locations. In particular, we are

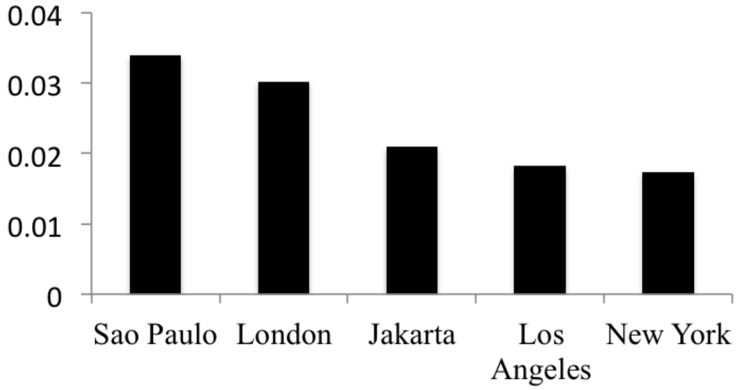


Figure 4.3: Top-5 locations with most hashtags

interested in understanding: (i) what is the worldwide distribution of hashtags? (ii) does distance between two locations influence which hashtags they adopt? and (iii) does distance between two locations influence when they will adopt these hashtags?

4.4.1 Location Distribution

We first examine the distribution of hashtags across the 4,946 unique locations represented in the dataset, as shown in Figure 4.2. The distribution of hashtags occurring in locations ordered by their rank (in terms of number of occurrences) decreases exponentially with increasing rank, meaning that the distribution of hashtags in various locations is very uneven. But, focusing on just the top-200 locations (as shown in the inset plot in Figure 4.2), we see that though the decrease in occurrence is exponential, it is small compared to the drop that we see for all locations in the larger figure, indicating the presence of locations that generate high but relatively the same number of hashtags.

The top-5 locations by their rank are shown in Figure 4.3. While Sao Paulo claims close to 3.4% of all hashtags and no US city occurs in the top-3 positions, when aggregating locations by country we observe that the US has close to a 40%

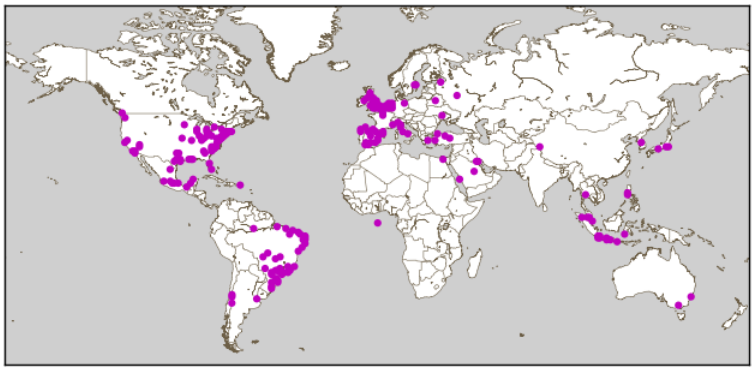


Figure 4.4: Top-200 locations with the most hashtags.

share followed by Brazil with 6% and the UK with 5%.

Although the US dominates, if we extend to the top-200 most prevalent locations, we see in Figure 4.4 the global footprint of hashtags covering most of the major densely populated cities in the world (sans China).

4.4.2 Relationship between Locations

Given the global nature of hashtags, we next examine the relationship between locations in terms of hashtag adoption. We consider two approaches that consider the distance between location pairs – one based on the fraction of hashtags shared between locations; the other based on the adoption time lag between locations. In both cases we measure the distance between locations using the Haversine distance function, which accounts for the effects of the Earth's spherical shape in finding distances between points.[†] In essence, the Haversine maps from latitude-longitude pairs to distance: $\mathcal{D} : \mathbb{R}^2 \times \mathbb{R}^2 \rightarrow \mathbb{R}$.

Hashtag Sharing vs Distance: We first seek to understand the relationship of the distance between locations on the commonality of hashtags adopted in locations.

[†] For a fuller treatment, we refer the interested reader to http://en.wikipedia.org/wiki/Haversine_formula

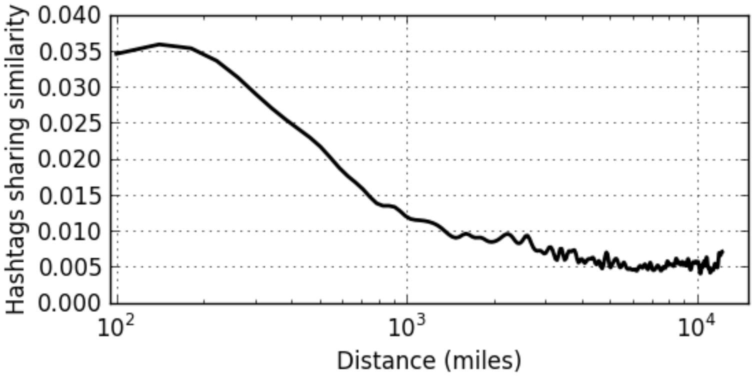


Figure 4.5: Hashtag sharing similarity vs distance.

To what degree does distance impact whether a hashtag is shared between two locations? Given two locations, we measure their hashtag "similarity" using the Jaccard coefficient between the sets of hashtags observed at each location:

$$\text{Hashtag Similarity}(l_i, l_j) = \frac{H_{l_i} \cap H_{l_j}}{H_{l_i} \cup H_{l_j}}$$

where recall H_l is the set of unique hashtags observed in l. Locations that have all hashtags in common have a similarity score of 1.0, while those that share no hashtags have a score of 0.0. The relationship between hashtag similarity and distance is plotted in Figure 4.5. We see a strong correlation, suggesting that the closer two locations are, the more likely they are to adopt the same hashtags. As distance increases, the hashtag sharing similarity drops accordingly. Much of this distance-based correlation can be explained by issues of language, culture and other common interests shared between these locations. For example, we see strong similarities in hashtags between English-speaking parts of Western Europe and the United States; and between Portuguese-speaking parts of Brazil and Portugal.

Hashtag Adoption Lag vs Distance: While locations that are near are more

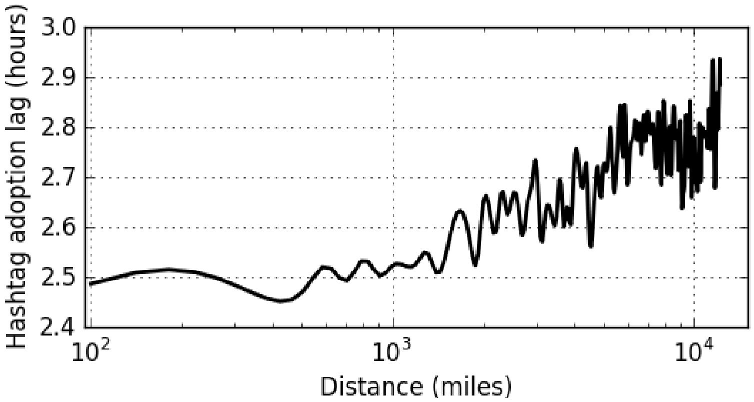


Figure 4.6: Hashtag adoption lag vs distance

likely to share hashtags, are they also more likely to adopt hashtags at the same time? We next measure the impact of distance on hashtag adoption lag between two locations. Locations that adopt a common hashtag at the same time can be considered as more temporally similar than are two locations that are farther apart in time (with a greater lag). Letting t_l^h be the first time when hashtag h was observed in location l, we can define the hashtag adoption lag of two locations as:

$$\text{Adoption Lag}(l_i, l_j) = \frac{1}{|H_{l_i} \cap H_{l_j}|} \sum_{h \in H_{l_i} \cap H_{l_j}} |t_{l_i}^h - t_{l_j}^h|$$

where the adoption lag measures the mean temporal lag between two locations for hashtags that occur in both the locations. A lower value indicates that common hashtags reach both locations around the same time. We see in Figure 4.6 a relatively flat relationship up to ~500 miles, then a generally positive correlation, suggesting that locations that are close in spatial distance tend also to be close in time (e.g., they adopt hashtags at approximately the same time). Locations that are more spatially distant tend to adopt hashtags at greater lags with respect to each other.

4.4.3 Summary

Our observations in this section indicate that hashtags are fundamentally a global phenomenon, with locations all across the world participating in the sharing of this type of social media. However, we have also confirmed that the physical distance between locations is a strong constraint on the adoption of hashtags, both in terms of the hashtags shared between locations and in the timing of when these hashtags are adopted.

4.5 Hashtag Propagation

Based on the observations in the previous section, we now focus on the characteristics of hashtag propagations across the globe. We examine the spatio-temporal properties of individual hashtags to explore questions like: To what degree are hashtags a local phenomenon? Does the number of occurrences of hashtag impact its global spread? Can we characterize the spatial properties of local and global hashtags?

4.5.1 Spatial Properties of Hashtag Propagation

Previous studies of the geographic scope of social media and web resources have typically adopted two types of measures: one considering the intensity of focus and one considering the uniformity of this interest. Similarly, we adopt two measures (similar to ones for studying YouTube videos in [9]): *hashtag focus* and *hashtag entropy*, plus a third measure called the *hashtag spread*.

For every hashtag ($h \in H$) and location ($l \in L$) pair, if we let O_l^h be the set of all occurrences of h in l, then the probability of observing hashtag h in location l is defined as:

$$P_l^h = \frac{O_l^h}{\sum_{l \in L}\{O_l^h\}}$$

Then the *hashtag focus* for hashtag h is:

$$\mathcal{F}^h = \max_{l \in L} P_l^h$$

which is simply the maximum probability of observing the hashtag at a single location. The location at which the probability is maximum is called the *hashtag focus location*. As a hashtag propagates, intuitively its focus will reduce as the hashtag is observed at multiple locations. The more local a hashtag is, presumably the higher its focus will be as well. Note that we additionally denote the focus measured over an interval t (rather than over the entire dataset) as $\mathcal{F}^h(t)$.

The *hashtag entropy* is defined as:

$$\mathcal{E}^h = -\sum_{l \in L} P_l^h \log_2 P_l^h$$

which measures the randomness in spatial distribution of a hashtag and determines the minimum number of bits required to represent the spread. A hashtag that occurs in only a single location will have an entropy of 0.0. As a hashtag spreads to more locations, its entropy will increase, reflecting the greater randomness in the distribution. Like focus, we can additionally denote the entropy measured over an interval t (rather than over the entire dataset) as $\mathcal{E}^h(t)$.

While focus and entropy provide insights into a hashtag's locality, they lack explicit consideration for the distance a hashtag has traveled. For example, consider two hashtags – one distributed equally between Austin and Dallas, and another one equally distributed between Los Angeles and New York. The focus of both hashtags is 0.5 and their entropy is 1. Hence, to measure the greater "dispersion" of the

LA-NY hashtag, we define the *hashtag spread* of hashtag h as:

$$\mathcal{S}^h = \frac{1}{|O^h|} \sum_{o \in O^h} \mathcal{D}(o, G(O^h))$$

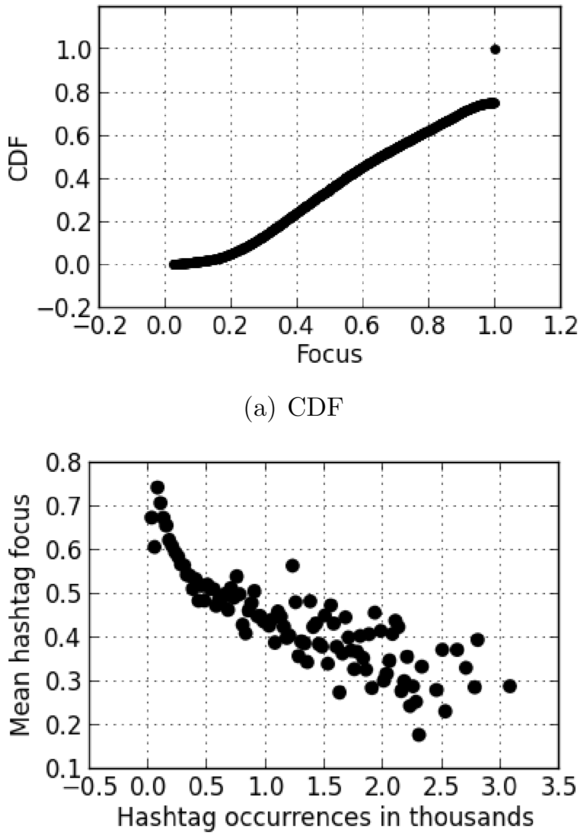
which measures the mean distance for all occurrences of a hashtag from its geographic midpoint. Here, G is the geographic midpoint[‡] for a set of occurrences, which is similar to calculating the midpoint on a plane for a set of 2-dimensional points, but as in the case of Haversine distance, the geographic midpoint is calculated by considering the effects of Earth's spherical shape. A local hashtag with many occurrences close to its midpoint will yield a small spread, while a global hashtag with occurrences relatively far from its center will yield a larger spread.

4.5.2 Local versus Global: Measuring Focus, Entropy, and Spread

Using these three spatial properties, we now analyze the properties of hashtag propagations.

Measuring Hashtag Focus: We begin by considering the focus values of hashtags. The cumulative distribution for focus values of hashtags is shown in Figure 4.7(a). We observe that the distribution is nearly linear, meaning that the focus values for hashtags are uniformly distributed. We also notice that most hashtags are concentrated in one location. Specifically, around 50% of hashtags derive at least 50% of their postings from a single location. In addition, as indicated by the single dot at CDF = 1.0, about a quarter of all hashtags are observed in a single location only. Continuing this look at hashtag focus, we next plot the relationship between the number of occurrences of a hashtag and its focus in Figure 4.7(b). As can be expected, we observe that hashtags with a few occurrences have a high focus (meaning that

[‡] http://www.geomidpoint.com/



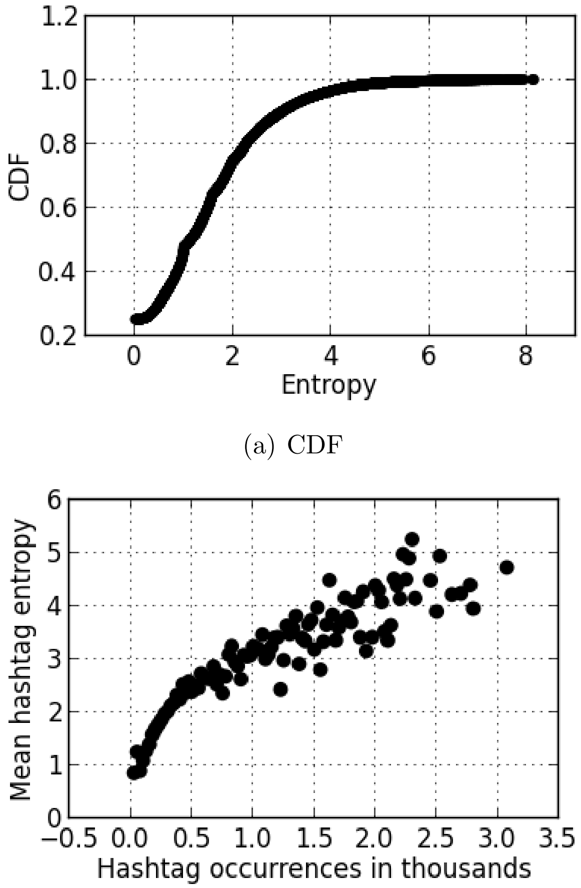
(a) CDF

(b) Mean hashtag focus

Figure 4.7: Focus: Around 50% of hashtags accumulate at least 50% of their postings from a single location.

these low-intensity hashtags tend to occur primarily in a single location), whereas an increasing number of occurrences corresponds to a decrease in the focus of the hashtag. Together, these results suggest that many hashtags correspond to either local events (e.g., #momentoschampions, #nyadaauditions) or geographically compact networks of friends. But as hashtags become more popular they tend to spread to more locations. That is, it is unlikely for a popular hashtag to be constrained to a handful of locations; there is spillover from one location to the next.

Measuring Hashtag Entropy: To further explore this spatial distribution, we

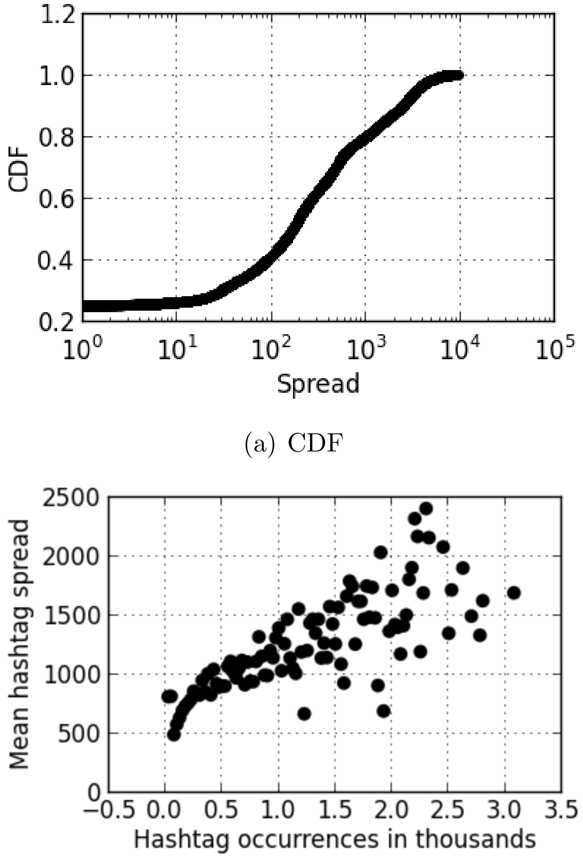


(a) CDF

(b) Mean hashtag entropies

Figure 4.8: Entropy: Almost 20% of hashtags are confined to a single location, but hashtags begin to spread as they become popular.

next consider the entropy of hashtag propagations. Recall that an entropy of zero for a hashtag indicates that it was posted from one (2^0) location only, while, for example, an entropy value of two indicates a hashtag propagated almost equally to four (2^2) locations. The cumulative distribution of entropy in Figure 4.8(a) shows that about 25% of hashtags are concentrated in a single location and that the majority of hashtags propagate to at most two locations. On the flipside, however, we do see that hashtags with many occurrences tend to spread to many locations, as seen by the



(a) CDF

(b) Mean hashtag spread

Figure 4.9: Spread: 50% of hashtags have a spread less than 400 miles; 25% of hashtags have a spread greater than 1000 miles.

increasing entropy versus the number of hashtag occurrences in Figure 4.8(b) (and the decreasing focus values, as we observed in Figure 4.7(b)). As a hashtag becomes popular it tends to spread to newer locations and this in turn makes it more popular. These results show that the majority of hashtags have a narrow base of geographic support, but that one of the keys to popularity is a broad geographic footprint. This is intuitively sensible, but important to confirm in practice.

Measuring Hashtag Spread: While focus and entropy provide insights into a hashtag's locality, neither directly measures the geographic area over which a hashtag

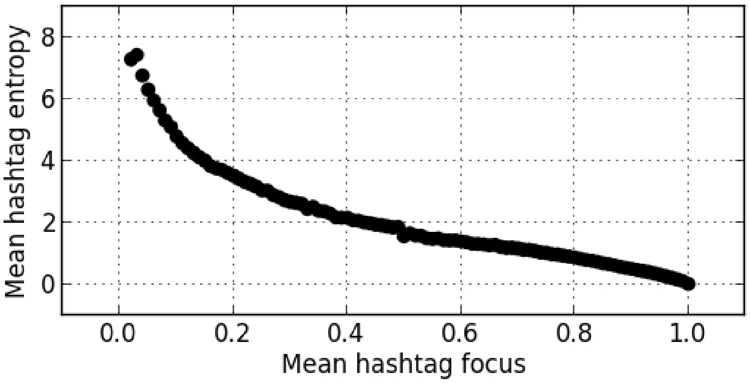
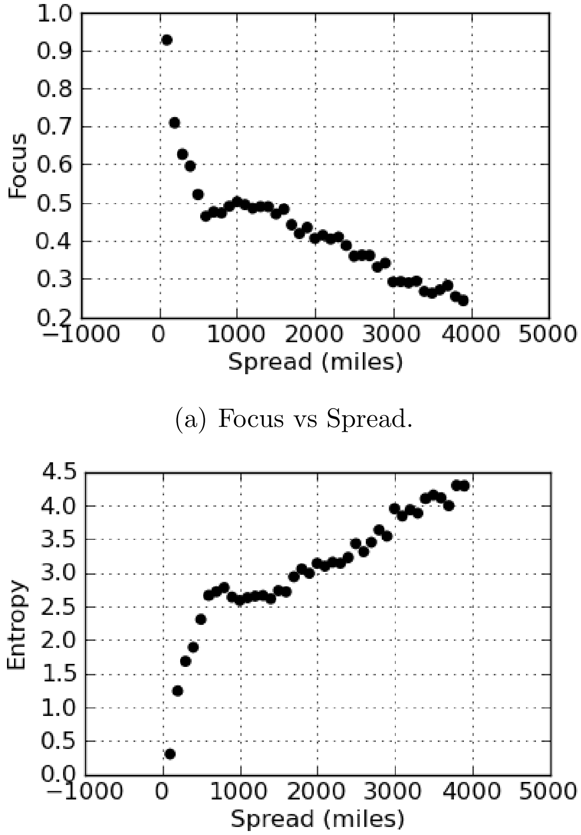


Figure 4.10: Entropy versus focus.

propagates. Using hashtag spread, we see in Figure 4.9(a) that about a quarter of hashtags have a spread of zero since they were observed in only location. In addition, we observe that most hashtags have a small spread, with almost 50% of hashtags having a spread less than 400 miles. However, we do observe that around 25% of hashtags have a spread greater than 1000 miles. We next plot the correlation between number of occurrences of a hashtag and its spread in Figure 4.9(b). Consistent with the findings over focus and entropy we observe that an increasing number of occurrences is coupled with a larger spatial footprint.

Direct Comparison of Spatial Properties: We now turn to directly comparing the focus, entropy, and spread values for our hashtags. We begin by plotting the mean hashtag focus on the x-axis versus the mean hashtag entropy on the y-axis, as shown in Figure 4.10. Local hashtags – with a high focus and a low entropy – are located in the bottom-right of the figure; global hashtags – with a low focus and a high entropy – are located in the top-left of the figure.

The correlation between spread and our two other spatial properties – focus and entropy – is shown in Figure 4.11. As expected, an increasing spread results in a decreasing focus because as a hashtag spreads it occurs in more locations which in turn reduces the overall focus. For similar reasons we observe an increase in entropy with increasing spread.

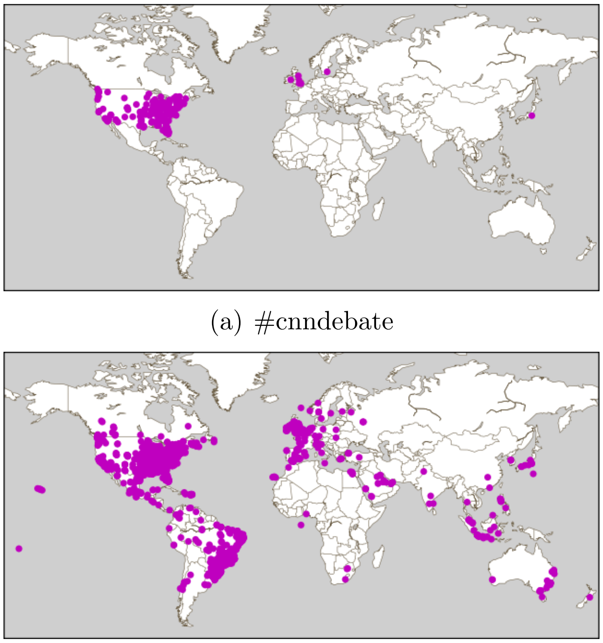


(a) Focus vs Spread.

(b) Entropy vs Spread.

Figure 4.11: Correlation between spatial properties and spread.

We also observe that in Figure 4.11(a), there is a steep drop in focus for the first 700 miles, followed by a region of almost uniform focus until about 1600 miles and finally a region of decreasing focus until 4000 miles. The initial steep drop of focus indicates that the locations that are adopting hashtags are spatially close to each



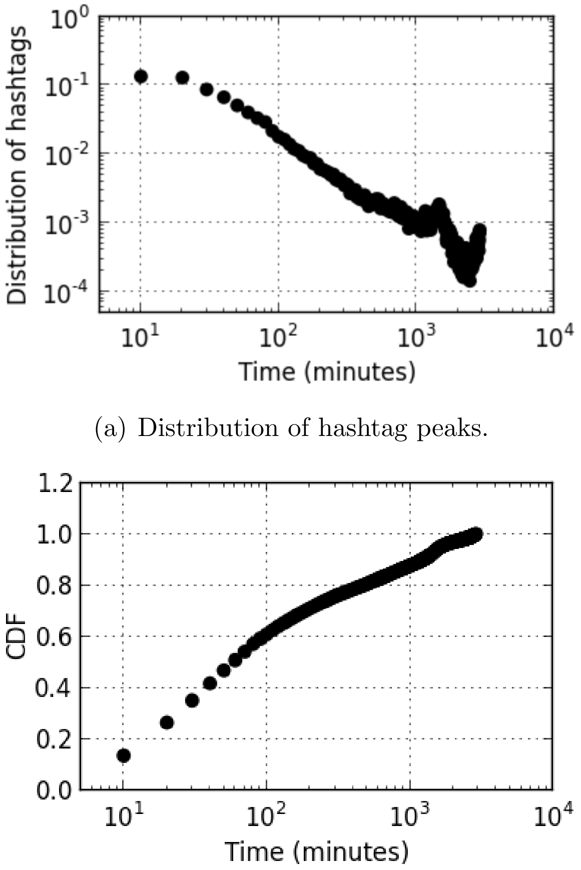
(a) #cnndebate

(b) #ripstevejobs

Figure 4.12: Example of hashtag spread.

other. On a map, the spatial distribution of these hashtags would look like a tight cluster of dots in a small region. The next region where the focus remains almost the same while the spread increases corresponds to hashtags that are spatially well distributed but the majority of hashtags are being produced by a single location. On a map the spatial distribution for these hashtags would have dots spread over a wide region as in Figure 4.12(a), but with only a few of those dots generating the majority of hashtags. Finally, the third region corresponds to globally distributed hashtags like the one shown in Figure 4.12(b). We see similar behavior when we plot entropy against spread as shown in Figure 4.11(b): a steep increase in entropy for the first 700 miles, then a region until about 1600 miles with uniform entropy and finally a region of increasing entropy until 4000 miles.

In summary, most hashtags are essentially a local phenomenon, as indicated by



(a) Distribution of hashtag peaks.

(b) CDF for hashtag peaks.

Figure 4.13: Hashtag peak analysis.

the on-average high focus, low entropy, and small spread. But as a hashtag becomes more popular, we see a decrease in focus and an increase in entropy and spread, all hallmarks of global impact. Based on the analysis in this section, we identify three broad categories of hashtags:

- **Local Interest [60% of all hashtags]**: These hashtags have a spread range from 0 to 700 miles. They have a high focus with median of 0.79 and low entropy of 1 bit. Example local interest hashtags include *#volunteer4betterindia*, *#ramadanmovies*, and *#onceuponatimeinnigeria*.

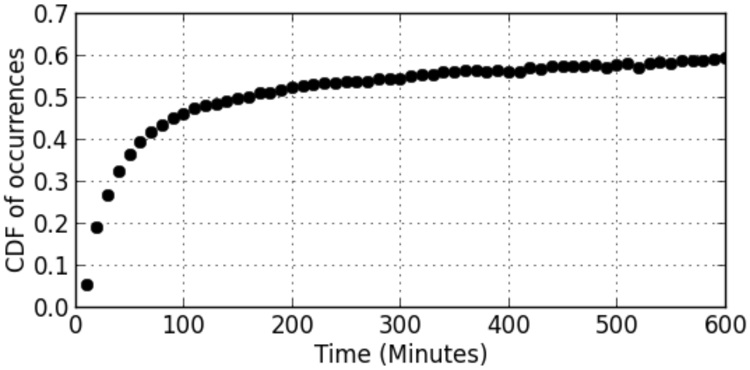


Figure 4.14: CDF of occurrences with time.

- **Regional and Event-Driven [15% of all hashtags]**: These hashtags have a spread range from 700 to 1000 miles. They have a median focus of 0.44 and entropy of 3 bits. Example regional and event-driven hashtags include#cnbcdebate, #iowadebate, etc.

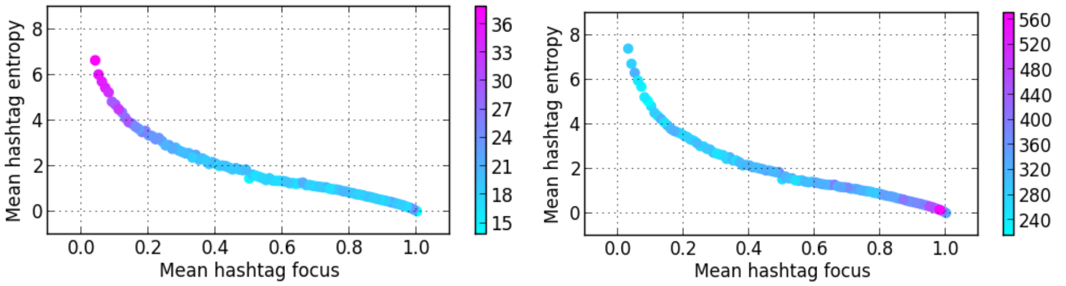
- **Worldwide Phenomena [25% of all hashtags]**: These hashtags have a spread range from 1000 to 4000 miles. These are mostly global hashtags which have low focus with median of 0.28 and entropy of 4 bits. Example worldwide phenomena hashtags include *#britneyvmas*, *#yearof4*, *#timessquareball*.

4.5.3 Slow versus Fast: Peak Analysis

We next augment our analysis by considering, in addition to the spatial propagation of hashtags, the temporal characteristics of these hashtags. We begin this temporal analysis by studying *when* hashtags reach the peak of their propagation in terms of occurrences. For this study we focus on hashtags that reach their peak within the first two days after their first appearance. We see in Figure 4.13(a) the distribution of *peak times* across all hashtags. We find that around 20% of hashtags reach their peak within 20 minutes of their first appearance. The distribution of peaks falls exponentially after that. We also observe that about 60% of all hashtags reach their peak within the first 2 hours as shown in Figure 4.13(b). In addition we observe that on average hashtags accumulate more than 50% of their total occurrences in the first 2 hours of their propagation as shown in Figure 4.14.

But what are the differences between fast-peaking hashtags and slow-peaking ones? Do hashtags behave differently in terms of their spatial properties? To answer these questions, we consider two sets of hashtags – those that reach their peak within the first 30 minutes of their initial appearance and a second set consisting of slower hashtags that reach their peak between 4 and 10 hours of their initial appearance. To analyze the relationship between locality and peak times we plotted these sets of hashtags in Figure 4.15, with focus on the x-axis and entropy on the y-axis.

We observe that in the set of faster hashtags – which reach a peak within 30 minutes of their propagation – the local hashtags are much faster than the global ones (see Figure 4.15(a)). This observation is reversed in the set of slower hashtags, shown in Figure 4.15(b), where the global hashtags are relatively faster than the local hashtags. On closer inspection, we attribute this reversal to the motive or purpose of the hashtags. First, we observe that hashtags that peak slowly are mostly of anticipated events, like the hashtag "#mtvema" corresponding to the MTV music awards, while the hashtags that peak more quickly are those that are organically generated within Twitter and related to fun like "#childhoodmemories". Second, slower hashtags are not as dependent on social sharing within Twitter as compared to faster hashtags; for example, users may be aware about the MTV awards from multiple sources (TV, news, friends), while the hashtag "#childhoodmemories" is seen only by those on Twitter. This dependency on the network to spread makes local fast hashtags peak sooner than the global fast hashtags. The global slow hashtags peak sooner than the local slow hashtags since more people are aware about them



(a) Hashtags that peak during the first 30 minutes.

(b) Hashtags that peak between 4 and 10 hours

Figure 4.15: (Color) Comparing the spatial properties of hashtags that reach their peak quickly (a) and those that reach their peak more slowly (b). Local hashtags – with a high focus and a low entropy – are located in the bottom-right of each figure; global hashtags – with a low focus and a high entropy – are located in the top-left of each figure. Low peak values are in light blue; high peak values in magenta.

and they are not dependent on the network.

Based on this peak analysis, we group hashtags into three categories:

- **Fast [25% of all hashtags]**: These hashtags reach their peak within 30 minutes of their first appearance. We find that 65% of these hashtags are local, 15% of these hashtags are national or event driven and 20% are global.

- **Medium [20% of all hashtags]**: These hashtags reach their peak between 30 minutes and 10 hours after their first appearance. We find that 55% of these hashtags are local, 17% of these hashtags are national or event driven and 28% are global.

- **Slow [55% of all hashtags]**: These hashtags reach peak more than 10 hours after their first appearance. We find that 60% of these hashtags are local, 16% of these hashtags are national or event driven and 24% are global.

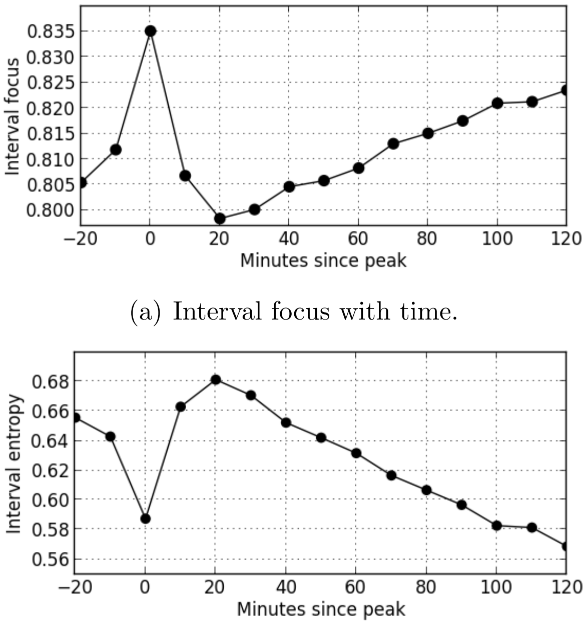
For all three peak-based categories we observe that the distribution of spatial categories is quite similar.

4.5.4 Patterns of Hashtag Propagation

We next zoom in on the spatial properties of hashtag propagation during the minutes pre- and post- peak. When hashtags peak, do they peak suddenly in different locations simultaneously or do they slowly accumulate a larger spatial footprint? What are the dynamics of their spatial properties as they become popular?

For this study, we divide each hashtag's lifecycle into equal length time intervals of 10 minutes. For each time interval, we compute the hashtag focus ($\mathcal{F}^h(t)$) and the hashtag entropy ($\mathcal{E}^h(t)$) over just that interval. We plot these interval-specific focus and entropy measures in Figure 4.16. First, compared to the aggregate characteristics across all hashtags – in which we find the median focus for all hashtags over their entire lifetime to be 0.57; for entropy, we find a median of 2 bits – here we see that the interval-based focus is even higher (greater than 0.80 in all cases) and the interval-based entropy is even lower (less than 1 bit in all cases). These higher focus / lower entropy results indicate that hashtags are *even more local* during each step of their propagation. To illustrate, in the aggregate we may find a hashtag that propagates only in locations in Texas. Compared to a global hashtag, it is certainly more local and its focus and entropy will reflect this. However, during its propagation, the Texas-based hashtag is even more local at each step; that is, it does not propagate over the entire state simultaneously but in stages, city by city. It might first become popular in Dallas, then in Austin, and so on.

Returning to Figure 4.16(a) and Figure 4.16(b), we observe that hashtags reach their lowest interval focus and highest interval entropy about 10-20 minutes after their peak. Rather than peaking with their most "global" footprint, hashtags instead reach this state *after* their peak. This result – that a peaking hashtag is actually more local than it ultimately will be – is seemingly counterintuitive. However, recall that in our



(a) Interval focus with time.

(b) Interval entropy with time.

Figure 4.16: Hashtags peak with their most "global" footprint 10-20 minutes after their peak

in our examination of the cumulative distribution of focus shown in Figure 4.7(a), we noted that almost 50% of hashtags accumulate more than 50% of their occurrences from a single location. With this in mind, we find that hashtags receive most of their occurrences from this single location during their peak explaining the spike in interval focus and the fall in interval entropy. In effect, this single location is "championing" a hashtag. In the 10-20 minutes after this peak period, other locations adopt the hashtag, resulting in a decrease in interval focus and an increase in entropy as the hashtags becomes more global. About 30 minutes after reaching peak, focus and entropy reverse, with focus increasing and entropy decreasing as the hashtag withdraws back to its original focus location.

In essence, hashtags are spread via a single location "championing" a hashtag initially, spreading it to other locations and then continuing to propagate it after it has become popular. In [9], the authors observed a similar pattern for YouTube videos which they called the "spray-and-diffuse" pattern. Our observations over hashtags suggest that this pattern may be a fundamental property of social media spread.

4.6 Summary

In this section, we have analyzed the spatio-temporal dynamics of social media propagation through a study of 2 billion geo-tagged Tweets. Our study has consisted of two key parts: (i) a study of the global footprint of hashtags and an exploration of the spatial constraints on hashtag adoption; and (ii) a study of three spatial properties of hashtag propagation – focus, entropy, and spread – and an examination of the spatial propagation of hashtags using these properties. We have found that hashtags are a global phenomenon, with locations all across the world. But the physical distance between locations is a strong constraint on the adoption of hashtags, both in terms of the hashtags shared between locations and in the timing of when these hashtags are adopted. We have also found that hashtags are mostly a local phenomenon with long-tailed life spans, but follow a "spray-and-diffuse" pattern [9] where initially a small number of locations "champion" a hashtag, make it popular, and the spread it to other locations. We have found both spatial and temporal locality as most hashtags spread over small geographical areas but at high speeds. The purpose of a hashtag and its global awareness determines how fast it will reach its peak. A hashtag representing a globally known event reaches its peak much faster than either locally-known events or hashtags spread purely within the network (e.g., #ifyouknowmeyouknow). Based on spatial and temporal categories we classified hashtags into different categories. In our continuing work we are interested in hashtag category specific analysis. We want to study how the temporal characteristics of hashtags may differ depending upon their spatial categories.

5. MODELING OF GEO BASED SOCIAL TRAILS*

In this section we seek to understand and model the global spread of social media. How does social media spread from location to location across the globe? Can we model this spread and predict where social media will be popular in the future? Toward answering these questions, we develop a probabilistic model that synthesizes two hypotheses that are at the extreme ends of explaining the nature of online information spread: (i) the spatial influence model, which asserts that social media spreads to locations that are close by; and (ii) the community affinity influence model, which asserts that social media spreads between locations that are culturally connected, even if they are distant. In addition, to this we develop another model that is in the middle of these two extreme models and blends the two models. Based on the geospatial footprint of 755 million geo-tagged hashtags spread through Twitter, we evaluate these models at predicting locations that will adopt hashtags in the future. We find that distance is the single most important explanation of future hashtag adoption since hashtags are fundamentally local. We also find that community affinities (like culture, language, and common interests) enhance the quality of purely spatial models, indicating the necessity of incorporating non-spatial features into models of global social media spread.

5.1 Introduction

As we discussed earlier, users generate and consume a great deal of content on the Internet every day in the form of videos, blogs, tweets, and so on. As users consume

*Part of this section is reprinted with permission from "Spatial Influence vs. Community Influence: Modeling the Global Spread of Social Media" by Krishna Y. Kamath, James Caverlee, Daniel Sui and Zhiyuan Cheng, 2012. Conference on Information and Knowledge Management. 21st. Copyright 2013 by Association for Computing Machinery (ACM).

and share this content, some of it tends to gain traction and become popular resulting in viral videos, trending hashtags, popular blogs, and so forth. These phenomena have attracted a considerable amount of recent research to study the dynamics of the adoption of social media [7, 37, 45, 47, 64].

Of particular importance is the geospatial spread of social media. For example, how did videos captured on smartphones during the Arab Spring spread across the globe? Are there key locations that promoted the spread of these videos? As the Arab Spring has become increasingly part of the US's social consciousness, do we see key US locations impacting the propagation of videos today? Answering these questions is extremely challenging, and so as a beginning step we study in this section the dynamics of social media adoption across geographical locations. Concretely, we formalize the problem of predicting the global spread of social media as the *location subset selection problem*. That is, as a particular item (e.g., video, image) begins to propagate can we predict the locations where it will soon become popular? For example, observing a video that is gaining traction in Qatar, can we predict locations in Europe where the video is soon going to become popular?

Previous work in the area of information (content) diffusion and influence propagation have tended to focus on the pathways of diffusion through social and information networks, e.g., [32, 42, 43, 44, 46, 78]. Complementary to these efforts, we focus on the geospatial connections that impact the spread of social media, and so we abstract from the interaction network layer to consider fine-grained locations and their connections to other locations. Towards modeling the global spread of social media, we develop a probabilistic model that synthesizes two hypotheses that are at the extreme ends of explaining the nature of online information spread:

- **Distance matters**. As encapsulated by Tobler's first law of geography [72] which asserts that all things being equal, closer places are more alike, whereas distant places are more unalike. In the context of social media spread, Tobler's first law of geography would suggest that locations that are close to each other should be more likely to adopt similar online behaviors (e.g., viewing a YouTube video, posting the same hashtag).

- **"Distance is dead"** [10, 68]. The second hypothesis claims that since online interactions are freed from geospatial constraints, mere proximity is no guarantee toward adopting similar online behavior. In this setting, long-distance links formed through common online community may be more predictive. For example, tech communities in Austin, San Francisco, and Seattle may be tightly linked through their common interest in similar YouTube videos, whereas more geographically close locations may share little in common.

Based on the first hypothesis, we develop the *spatial influence model*, which asserts that the adoption of a particular user activity in a nearby location has a stronger influence on a target location than whether that same activity was adopted at a more distant location. In other words, distance matters. Based on the second hypothesis, we develop the *community affinity influence model*, which asserts that locations that share a similar community affinity, regardless of distance from each other, are more likely to influence one another. While there are many ways to measure community affinity, we propose two methods: (i) the first considers communities to be close to each other if they share similar activities regardless of *when* they adopt these activities, for example tech communities in Austin and San Francisco reading similar articles on thehackernews.com; and (ii) the second considers communities close to each other if they tend to adopt similar activities in sync, like a video becoming popular in New York and Boston around the same time. Note that both the spatial influence model and the community affinity influence model are developed completely orthogonal to the underlying social network and are based solely on the geospatial distribution of user activities, meaning that estimating flows of influence from one person to another are not necessary. In addition, to this we develop another model that is in the middle of these two extreme models and blends the two models [61, 57]. We test these models in the context of the geospatial footprint of 755 million geo-tagged hashtags spread through Twitter. We find that while the spatial influence model has a higher impact than the community affinity influence model in predicting the spread, its combination with the community affinity influence model gives the best performance, suggesting that both distance and community are key contributors to social media spread.

The rest of the section is organized as follows. We start by describing related works in Section 5.2. In Section 5.3, we describe our dataset and measure geo-spatial properties of social media propagation. In Section 5.4, we formally define the location subset selection problem and present the spatial influence and community affinity models. Finally, in Section 5.5, we define the metrics to compare these models and evaluate the performance of these models before concluding in Section 5.6.

5.2 Related Work

Our work presented here builds on two lines of research: Twitter information diffusion and geo-spatial analysis of social media.

Information Diffusion on Twitter: There have been several papers studying the general properties of Twitter as a social network and in analyzing information diffusion over this network [37, 45, 46, 78]. Continuing in this direction most papers related to hashtags have focused their attention on understanding the propagation of hashtags on the network. For example, in [64] the authors studied factors for hashtag diffusion and found that repeated exposure to a hashtag increased the chance of it being reposted again, especially if the hashtag is contentious. An approach grounded in linguistic principles has been to study the property of hashtag creation, use, and dissemination in [17]. In related research, approaches based on linear regression have been used to predict the popularity of hashtags in a given time frame in [73]. Because of the semantic nature of hashtags and the variety of ways it is used to convey information about a tweet, there have been some papers which have used hashtags to solve problems like sentiment detection [20], topic tracking on twitter streams [48], and so forth.

Geo-spatial Analysis of Social Media: The emergence of location-based social networks like Foursquare, Gowalla, and Google Latitude motivated large-scale geo-spatial analysis [67, 56]. Some of the earliest research related to geo-spatial analysis of web content were based on mining geography specific content for search engines [22]. More recently in [4] the authors analyzed search queries to understand the spatial distribution of queries and understand their geographical centers. On Twitter, geo-spatial analysis has focused on inferring geographic information from tweets like predicting user locations from tweets [12] and spatial modeling to geolocate objects [19]. Similar analysis to infer user's location on Facebook based on their social network has been studied in [5]. A recent paper dealt with the spatial analysis of YouTube videos [9] . In this work the authors were able to observe the highly local nature of videos based on the propagation patterns of YouTube videos.

5.3 Measuring the Geospatial Properties of Social Media

In this section we first present notation for measuring social media spread with an eye toward developing models of this spread. Then we highlight the experimental setting – Twitter-based hashtags – and examine the geospatial properties of hashtag spread. Our goal is to study questions like: Does distance impact whether social media (hashtags, in this case) is shared between two locations? Does distance impact the timing of hashtag adoption? How predictable is the spread of a hashtag over a geographic area? Do early observations indicate whether a hashtag will spread compactly or be widely diffused over a large spatial area?

5.3.1 Preliminaries

Let M be the set of user activities of interest – for example, an activity could correspond to a click on a web link, a view of a Web video, sharing of a link on Facebook, posting a particular hashtag on Twitter, and so on. Suppose we have divided the globe into a set of distinct locations L (say by overlaying a mesh dividing the globe into squares of 0.001 degrees latitude by 0.001 degree longitude). Every activity is associated with some subset of locations in which the activity has been observed. For example, based on the IP address, a view of a Web video can be traced back to an approximate latitude and longitude. Similarly, many social media services and smartphones support GPS-enabled tagging of user activities. By discretizing time into regular intervals (say, into 5 second increments), we can express the set of occurrences of an activity $m \in M$ in a particular location $l \in L$ at time t as $o_l^m(t)$. For example, o_l^m may represent 10 clicks of a Web video m in the past minute, where each click originates in a particular neighborhood l.

Now, suppose we have observed all occurrences of an activity up to some critical time t_s. Then we can define the set of **observed occurrences** (O_l^m) of m at a single location l as:

$$O_l^m = \bigcup_{t=0}^{t_s} o_l^m(t) \tag{5.1}$$

and the **total observed occurrences** set O^m across all locations in L as:

$$O^m = \bigcup_{l \in L} O_l^m$$

We denote the set of unique hashtags observed in l as M_l.

5.3.2 Experimental Setting: Hashtags

To measure the geospatial properties of social media, we focus our attention on one type of globally observed user activity – the posting of hashtags on Twitter. Twitter hashtags are prefixed with a # and mostly serve as tags to the corresponding tweet. Users tag their tweets for different purposes. For example, some are event driven like #ripstevejobs, and #fukushima, while some are mostly for fun like #bestsportsrivalry and #ifyouknowmeyouknow.

We collected a sample of around ~755 million geo-tagged tweets containing ~10 million unique hashtags from Twitter using the Twitter Streaming API from February 1 to November 30, 2011. Each tweet in this sample is tagged with a latitude and longitude indicating the location of the user at the time of the posting. All $<$ `hashtag`, `time`, `latitude`, `longitude` $>$ tuples corresponding to a particular hashtag are considered as a single activity of interest. Together all hashtags give us the set of all activities M.

We round latitudes and longitudes to their nearest tenth values, which overlays a mesh dividing the globe into locations (L). To avoid sparsely represented hashtags, we consider only hashtags with at least 5 occurrences in a location and consider only hashtags with at least 250 total occurrences across all locations. Since some hashtags may have begun their Twitter life before the first day of our sample (February 1) while others may have continued on after the last day (November 30), we consider

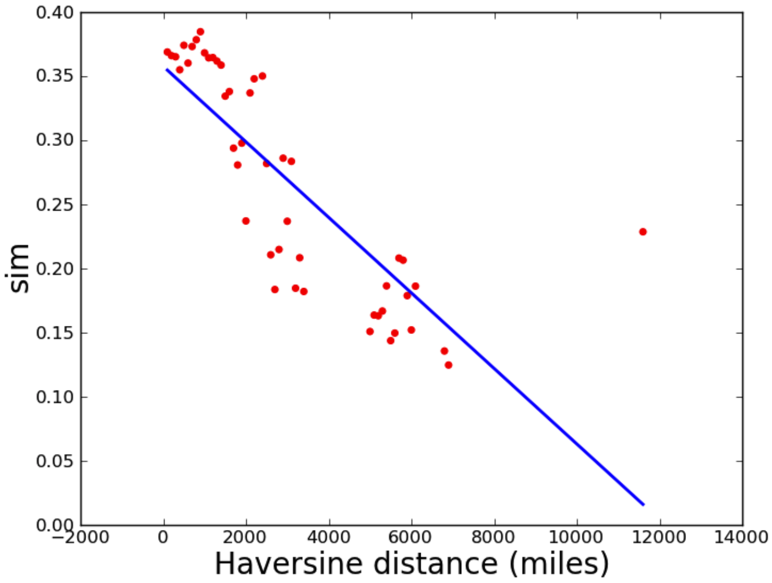


Figure 5.1: This figure shows the correlation between location similarity and distance. We see that similarity between location decreases with increasing distance.

both February and November as buffer months. Hence, we capture the full lifecycle of hashtags starting on or after March 1 and ending by October 31, which focuses our study to hashtags which have both their birth and death within the time of study (and as a result, removes cyclical hashtags like "#ff" and "#nofollow"). We additionally divide the set of all hashtags into two sets: a training set based on hashtags from March to August; and a test set based on September to October. Hashtags that start in training but continue into test are ignored. In this way, the training set contains 1466 complete hashtag propagations and the test set contains 515.

5.3.3 Geospatial Properties of Hashtags

Toward informing the development of models of social media spread, we study three geospatial properties of hashtags: (i) sharing versus distance, (ii) adoption lag versus distance, and (iii) the predictability of spread.

Hashtag Sharing versus Distance: We first seek to understand the relationship of the distance between locations on the commonality of hashtags adopted in locations. Do we find that distance has no impact on whether a hashtag is shared between two locations? We define the distance between two locations using the Haversine distance, which is commonly used to measure the distance between locations based on the spherical shape of the Earth (as compared to Euclidian distance)[†]. In essence, the Haversine maps from latitude-longitude pairs to distance: $\mathcal{D} : \mathbb{R}^2 \times \mathbb{R}^2 \rightarrow \mathbb{R}$. $H : L \times L \rightarrow \mathbb{R}_{\geq 0}$.

Given two locations, we measure their hashtag "similarity" using the Jaccard coefficient between the sets of hashtags observed at each location:

$$sim_{hashtag}(l_1, l_2) = \frac{M_{l_1} \cap M_{l_2}}{M_{l_1} \cup M_{l_2}}$$

where recall M_l is the the set of unique hashtags observed in l. Locations that have all hashtags in common have a similarity score of 1.0, while those that share no hashtags have a score of 0.0. The relationship between hashtag similarity and distance is plotted in Figure 5.1. We see a strong correlation ($\rho = -0.8$), suggesting that the more distant two locations are, the less alike they are. We also note that, though the similarities are high for most location pairs that are close to each other, there are some location pairs (above the blue line) where this doesn't hold true. Presumably, these outliers are linked by some other factors (language, culture), which we shall explore in the community affinity model shortly.

Hashtag Adoption Lag versus Distance: We additionally can measure the lag between two locations by measuring how close in time did the two locations adopt the same hashtag. Locations that adopt a common hashtag at the same time are

[†] For a fuller treatment, we refer the interested reader to http://en.wikipedia.org/wiki/Haversine_formula

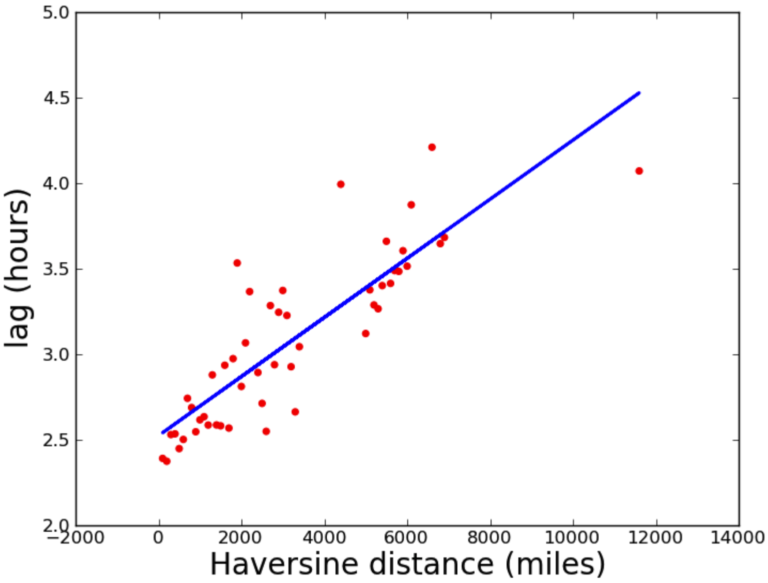


Figure 5.2: This figure shows correlation between hashtag adoption lag and distance. We see that adoption lag increases with increasing distance..

more similar (and have a smaller lag) than are two locations that are farther apart in time (with a greater lag). Letting M_l be the set of unique hashtags observed in l and t_l^m be the time of first occurrence of m at l, we can define the hashtag adoption lag of two locations as:

$$lag_{adoption}(l_1, l_2) = \frac{1}{|M_{l_1} \cap M_{l_2}|} \sum_{m \in M_{l_1} \cap M_{l_2}} |t_{l_1}^m - t_{l_2}^m|$$

where the adoption lag measures the mean temporal lag between two locations for hashtags that occur in both the locations. A lower value for this measure indicates that common hashtags appear to reach both the locations around same time. We see in Figure 5.2 a positive correlation ($\rho = 0.86$), suggesting that locations that are close in spatial distance tend also to be close in temporal distance (e.g., they adopt

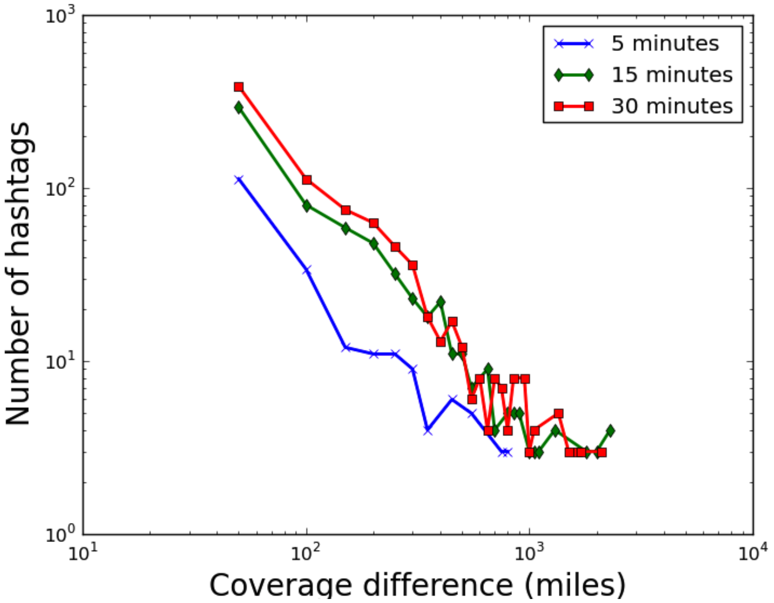


Figure 5.3: The figure shows comparison between early and late coverage for call hashtags. It indicates that most hashtags have a small difference between early and late coverage values.

hashtags at approximately the same time). Locations that are more spatially distant tend to adopt hashtags at much greater lags with respect to each other. As in the case of hashtag sharing, we see many location pairs having low lags despite being quite distant from each other, suggesting some other mechanism is at work.

Predictability of Spread: Finally, we measure the predictability of the "spread" of hashtag over a geographic area through its *coverage*. Coverage measures the mean Haversine distance for all occurrences of a hashtag from its geographic midpoint:

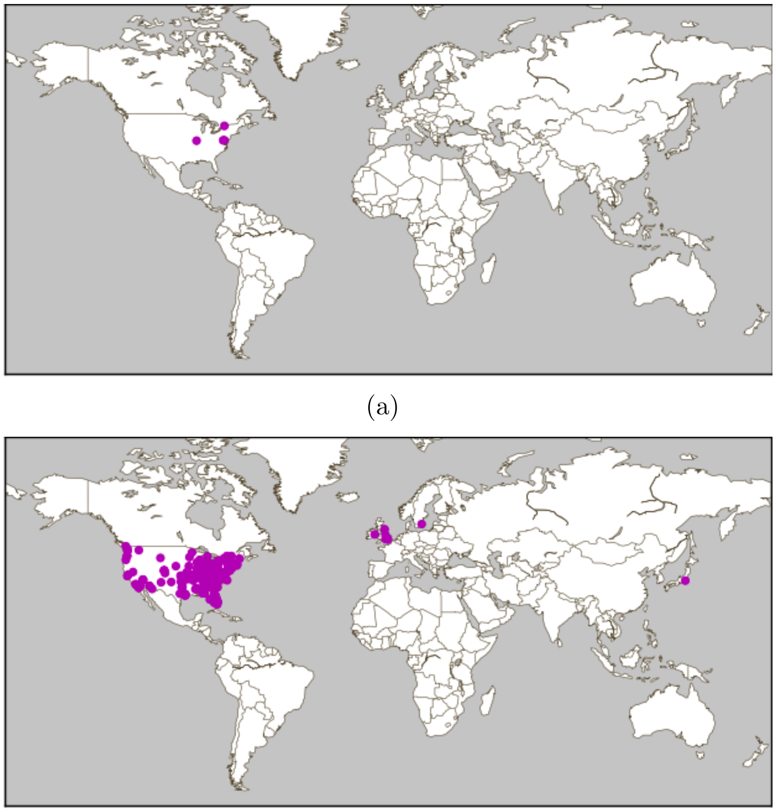
$$C(O^m) = \frac{1}{|O^m|} \sum_{o \in O^m} \mathcal{D}(o, G(O^m))$$

where we define the geographic midpoint[‡] for a set of occurrences as a function $G : O \rightarrow \mathbb{R}^2_{\geq 0}$, where the first dimension is the latitude and the second is the longitude of the midpoint. The calculation of geographic midpoint is similar to calculating the midpoint on a plane for a set of 2-dimensional points, but as in the case of Haversine distance, the geographic midpoint is calculated by considering the effects of Earth's spherical shape. A hashtag localized to a specific areas has a small coverage, while a universal hashtag has a larger coverage. To illustrate, consider the two hashtags #cnndebate and #ripstevejobs. Figure 5.4(b) shows the propagation of #cnndebate – corresponding to the Republican Presidential debate – after 2 hours. We see that the hashtag is mostly local to the United States and has a coverage of 743.32 miles. In contrast, Figure 5.5(b) shows the propagation of #ripstevejobs after 2 hours, resulting in a coverage of 3120.96 miles, indicating a global footprint.

To understand the predictability of spread, we measure the distribution of differences between the coverage for hashtags after they have completely propagated and coverage after the hashtag has propagated for a smaller time interval. For three initial periods – of 5 minutes, 15 minutes, and 30 minutes – we plot the difference between the coverage at this early time of a hashtag's propagation and the coverage after the completion of the hashtag's entire lifespan. We observe in Figure 5.3 that most hashtags have a small coverage difference, indicating that the final coverage of hashtag propagations can be accurately estimated early in its lifecycle. And the predictability of coverage increases as the length of the initial period increases (from 5 to 30 minutes); that is, as more evidence is accumulated over the beginning stages of a hashtag, the final coverage differs by less.

Continuing the example of #cnndebate and #ripstevejobs, we see in Figure 5.4 and Figure 5.5 that occurrences observed early in a hashtag's lifecycle (in this case,

[‡] http://www.geomidpoint.com/



(a)

(b)

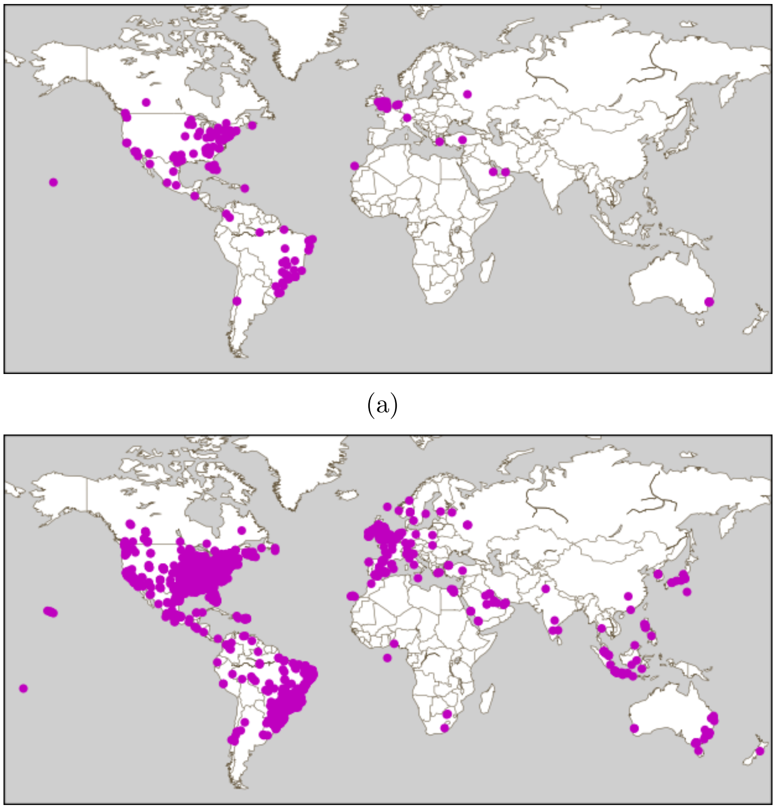
Figure 5.4: #cnndebate after 5 minutes (left) and 2 hours (right)

after just 5 minutes) are good indicators of later occurrences (in this case, measured after 120 minutes).

Based on these three geospatial properties, we observe:

1. In most cases, pairs of locations that are close to each other tend to share common hashtags and adopt them around the same time, compared with locations that are distant.

2. Many distant location pairs, though, exhibit similar patterns of "closeness" in that they share hashtags and have a low hashtag adoption lag, suggesting some additional factor is "bending space" to link the two locations.



(a)

(b)

Figure 5.5: #ripstevejobs after 5 minutes (left) and 2 hours (right)

3. Finally, the spread predictability analysis suggests that early occurrences of a hashtag are good indicators of the relative coverage of a hashtag's future spread (either compact or widely diffuse).

5.4 Modeling Hashtag Spread

Based on these observations, we next turn to the challenge of developing models of hashtag spread. Specifically we develop and evaluate the *spatial influence model* – in which nearby locations strongly influence hashtag adoption – and the *community influence model* – in which "similar", though perhaps distant, locations strongly influence hashtag adoption. The intuition behind both approaches is that locations influence each other, and that the future spread of a hashtag is guided by this mutual influence.

5.4.1 Problem Setting

To formalize the development of such hashtag spread models and to provide an experimental grounding for evaluating the quality of such models, we focus on the problem of selecting future locations that will adopt a hashtag based on the partial evidence of the hashtag's propagation up until that time. We call this the *location subset selection problem.* That is, as a particular social media begins to propagate can we predict the locations where it will soon arrive and become popular? For example, observing a video which is gaining traction in Qatar, can we predict locations in Europe where the video is soon going to become popular? The models developed for tackling this problem are an important and necessary step for supporting content localization, geo-advertising, fraud detection, and other social media analytics. It is particularly important that such models robustly predict the spread of social media while it is still developing (e.g., a video is going viral, a meme is becoming increasingly popular).

Recall the **total observed occurrences** set O^m across all locations in L ($O^m = \bigcup_{l \in L} O_l^m$) introduced in Section 5.3.1. In practice, these observed activities will vary by location. Early adopting locations may encompass many postings of a hashtag (or views of a Web video, ...), while later adopting locations will have few or no postings of a hashtag (or views of a video, ...), especially in the early moments of a hashtag's rise to popularity. Based on this state up to some time t_s, can we select some subset of locations $S \in L$ such that these locations are likely to observe many occurrences of the user activity.

For example, consider the three locations – New York, Dallas and Seattle – shown

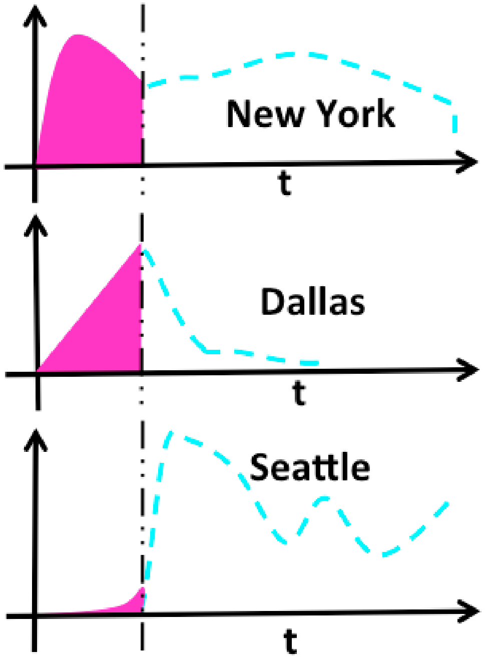


Figure 5.6: Based on the observed postings of a hashtag up to some time t_s (the vertical dotted line), can we predict which locations will post the most hashtags in the future?

in Figure 5.6 and suppose a particular hashtag has been posted from each location. Based on the observed hashtag postings up to some time t_s (the vertical dotted line), can we predict which locations will post the most hashtags in the future? Toward this goal, we can express the occurrences of the activity *after the critical time t_s* as the unknown future set of **unobserved occurrences**:

$$U_l^m = \bigcup_{t=t_s+1}^{\infty} o_l^m(t) \tag{5.2}$$

where U_l^m is the set of occurrences of m observed in location l after time t_s. We can additionally express the **total unobserved occurrences** set U^m across all locations in L as:

$$U^m = \bigcup_{l \in L} U_l^m$$

Together, the total occurrences of an activity throughout its lifetime is $O^m \cup U^m$. Now, suppose for some subset of locations $S \subseteq L$, we measure the count of the total unobserved occurrences of an activity in this subset as U_S^m:

$$U_S^m = \sum_{l \in S} |U_l^m|$$

We can then formulate the task of selecting the best k locations at some critical time t_s as the **location subset selection problem**:

Definition 5.4.1. ***(Location Subset Selection Problem)****: Given an integer k, the location subset selection problem for a user activity m at time t_s is the problem of predicting top-k locations which will have the highest number of unobserved occurrences for m.*

$$\mathcal{M}(m, L) = S_{t_s}^m = \underset{\{S \subseteq L \ | \ |S|=k\}}{\arg\max} \ U_S^m$$

where, $\mathcal{M} : M \times L^{|L|} \to L^k$, defined as subset selection model, takes a user activity and the set of all locations as input and returns a subset of locations of cardinality k.

The challenge for identifying the best choice of locations $S_{t_s}^m$ at time t_s is difficult because the future occurrences set for all locations, U^m, is available only after the complete evolution of the activity of interest. Hence, we must predict which locations are the best. Of course, determining the best choice of locations is simpler the longer the decision point is delayed (since many bursting and trending phenomenon will have run their course, saturating its locations), but of less value. The question is whether the best set of locations $S_{t_s}^m$ can be identified for some time t_s close to the activity's first observed occurrence.

5.4.2 Modeling Spread: Spatial Influence vs. Community Influence

With the problem statement in mind as well as our observations of the geospatial spread of hashtags, we now propose location influence based models for geo-spatial spread. The intuition behind our approach is that locations influence each other. And given a hashtag distribution, the future propagation of this hashtag is guided by this mutual influence between locations. The influence exerted by a location on another could be based either on proximity between locations or on the culture, language, and common interests shared by these locations. We measure this influence using an influence metric $\mathcal{I}^{l_i \to l_j}$ which has a range of $[0, 1]$ and represents the influence location l_i has on l_j such that the higher the value of this metric, then the greater is the influence exerted by l_i on l_j.

So given a hashtag m, the spread model for an influence metric $\mathcal{I}^{l_i \to l_j}$ is defined as:

$$\mathcal{M}_{\text{Spread}}(m, L) = \operatorname*{arg\,max}_{\{S \subseteq L \ | \ |S| = k\}} \sum_{l \in S} \left(P_l^m + \sum_{l_i \in L - l} P_{l_i}^m \cdot \mathcal{I}^{l_i \to l} \right)$$

where, $P_l^m = \frac{|O_{l_i}^m|}{|O^m|}$ is the probability of observing user activity m in l, estimated based on m's propagation until t_s and the expression within the parenthesis calculates the total effective influence exerted at this location to generate m. This concept is shown in Figure 5.7, where the location l_l gets influenced by all the locations and the effective influence on it is calculated as shown above. The spread model relies on the third observation that early occurrences of a hashtag are good predictors of future coverage. Hence, in this expression we use the probability of observing m in l to modify l's influence while calculating the effective influence. In this way the spread model, $\mathcal{M}_{\text{Spread}}$, selects a subset of the most influenced locations with the belief that

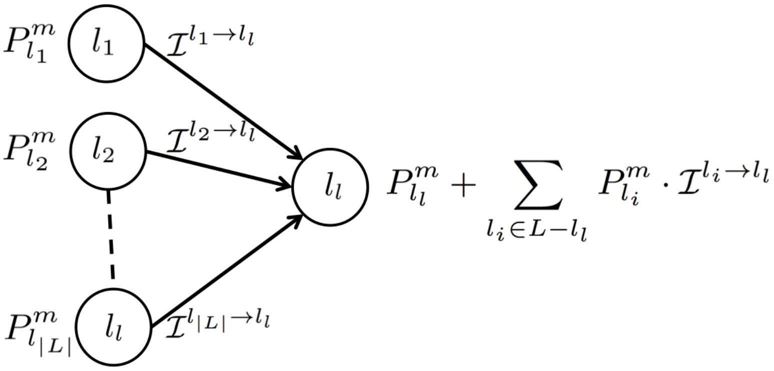

$P^m_{l_1}$
l_1
$\mathcal{I}^{l_1 \to l_l}$
$P^m_{l_2}$
l_2
$\mathcal{I}^{l_2 \to l_l}$
$P^m_{l_{|L|}}$
l_l
$\mathcal{I}^{l_{|L|} \to l_l}$
l_l
$P^m_{l_l} + \sum_{l_i \in L - l_l} P^m_{l_i} \cdot \mathcal{I}^{l_i \to l_l}$

Figure 5.7: General spatial influence model.

this influence will make these locations adopt hashtags in future.

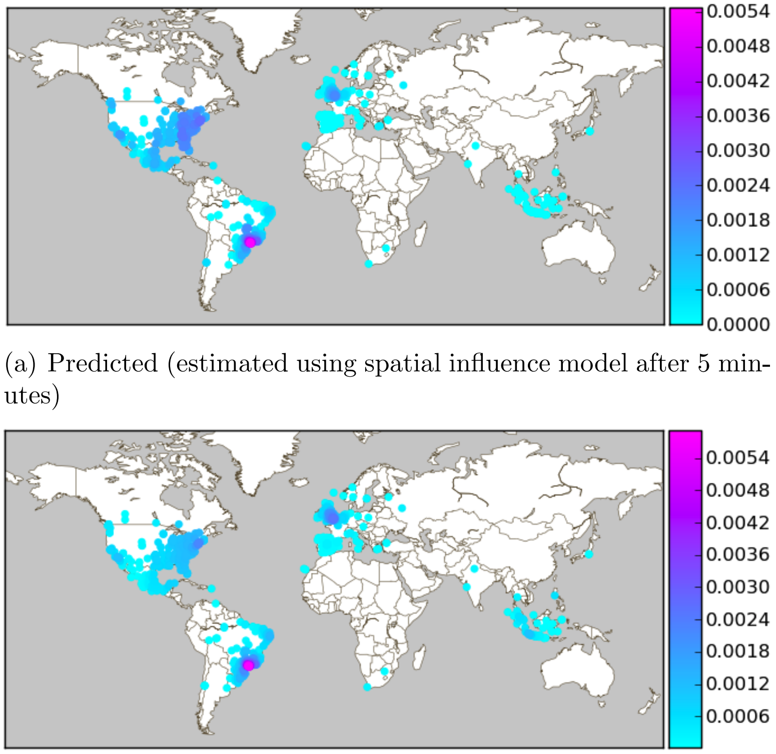
Using the spread model as framework, we now describe two general approaches – the spatial influence model and the community affinity model – that build on the observations made in Section 5.3.

5.4.2.1 Spatial Influence Model

The spatial influence model is based on our first observation in Section 5.3.3 that tells us that distance between locations influences what hashtags are shared and when they are shared. So, we define the spatial influence metric, $\mathcal{I}^{l_j \to l_i}_{\text{Spatial}}$, as:

$$\mathcal{I}^{l_j \to l_i}_{\text{Spatial}} = \frac{\alpha^{-H(l_i, l_j)}}{\sum_{l_i \in L} \alpha^{-H(l_i, l)}}$$

where, the numerator exponentially decays l_i's influence on l as a function of their Haversine distance and the denominator normalizes this influence so that $\sum_{l \in L} \mathcal{I}^{l \to l_i}_{\text{spatial}} = 1.0$. The parameter α controls the rate of influence decay. A higher value for α decreases influence from a point at a higher rate and a lower value for alpha (> 1.0) decreases influence at a lower rate. Using the this influence metric we define the



(a) Predicted (estimated using spatial influence model after 5 minutes)

(b) Actual (real distribution after 2 hours of propagation)

Figure 5.8: Example of using spatial influence model for #ripstevejobs

spatial influence model as:

$$\mathcal{M}_{\text{Spatial}}(m, L) = \underset{\{S \subseteq L \ | \ |S| = k\}}{\arg\max} \sum_{l \in S} \left(P_l^m + \sum_{l_i \in L - l} P_{l_i}^m \cdot \mathcal{I}_{\text{Spatial}}^{l_i \to l} \right) \tag{5.3}$$

To illustrate, consider an example of a hashtag that occurs only in Houston. Now given an option between Austin and San Francisco, the model as defined in (5.3) picks Austin since it is much closer to Houston than San Francisco.

A real world example of modeling propagations using the spatial influence model for the hashtag #ripstevejobs is shown in Figure 5.8. We predicted the future distribution of this hashtag using the spatial influence model based solely on its initial (first 5 minutes) distribution. The comparison between the predicted and actual distribution is shown in Figure 5.8(a) and Figure 5.8(b) respectively. We observe that the relative distribution (indicated by color) and its values (indicated by scale) are very close to each other.

5.4.2.2 Community Affinity Influence Model

Of course, distance is not the only factor that impacts the spread of a hashtag, as we observed in Section 5.3.3 (second observation). Hence, we now propose the community affinity influence models for capturing non-distance links between locations like culture, language, and common community interest. Concretely, we define two influence metrics to model community affinity based on their common usage of hashtags.

- **Transmitting Influence**: Using temporal proximity, we observe that if a hashtag is observed at a particular location, then it will soon be observed in other related locations as well. To model the degree to which a location can impact other locations temporally, we define the transmitting score, $\mathcal{T}$, as:

  $$\mathcal{T}_{l_j \rightarrow l_i} = \frac{|\{m \mid t^m_{l_j} > t^m_{l_i} \quad \forall m \in M_{l_i} \cap M_{l_j}\}|}{|M_{l_i}|}$$

  where, the numerator is the number of hashtags that occurred in l_1 before l_2. So, when all hashtags occurring in l_1 have occurred in l_2 and all before occurring in l_2, the transmitting score for l_1 transmitting a hashtag to l_2 - $P_t(l_2|l_1) = 1.0$. Using this we define the transmitting influence as:

  $$\mathcal{I}^{l_j \rightarrow l_i}_{\text{Trans.}} = \frac{\mathcal{T}_{l_j \rightarrow l_i}}{\sum_{l \in L} \mathcal{T}_{l \rightarrow l_i}} \tag{5.4}$$

  A value for $\mathcal{I}^{l_j \rightarrow l_i}_{\text{Trans.}}$ is in the range $[0, 1]$, with 0 indicating l_j doesn't transmit anything to l_i and 1.0 indicating l_j is the only location influencing l_i and it gets all of its hashtags after l_j.

- **Sharing Influence**: Similar to transmitting influence, we use content-related proximity to model the impact a location can have on nearby locations, using the sharing score:

  $$\mathcal{S}_{l_j \rightarrow l_i} = \frac{|M_{l_i} \cap M_{l_j}|}{|M_{l_i}|}$$

  This function measures the probability that l_i observes the same hashtags as l_j. Using this we define the sharing influence as:

  $$\mathcal{I}_{\text{Share}}^{l_j \rightarrow l_i} = \frac{\mathcal{S}_{l_j \rightarrow l_i}}{\sum_{l \in L} \mathcal{S}_{l \rightarrow l_i}} \tag{5.5}$$

  A value for $\mathcal{I}_{\text{Share}}^{l_j \rightarrow l_i}$ is in the range $[0, 1]$, with 0 indicating l_j doesn't share anything with l_i and 1.0 indicating l_j is the only location the influencing l_i and all hashtags that have occurred in l_i have occurred in l_j.

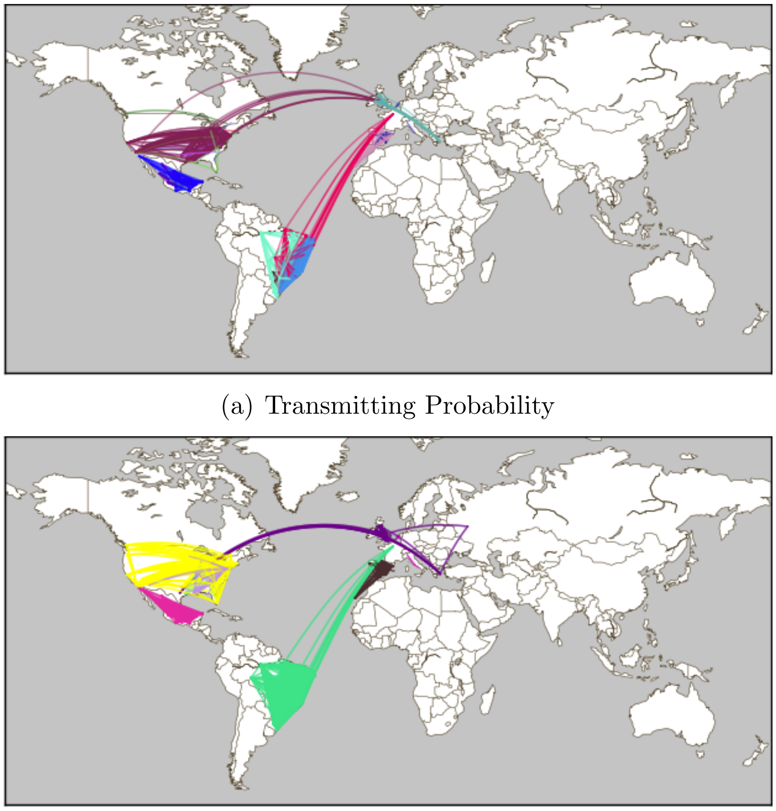
As in the case of the spatial influence model, we can use these two community affinity influence metrics to generate a model as:

$$\mathcal{M}_{\text{Trans.}}(m, L) = \underset{\{S \subseteq L \ | \ |S|=k\}}{\arg\max} \sum_{l \in S} \left(P_l^m + \sum_{l_i \in L-l} P_{l_i}^m \cdot \mathcal{I}_{\text{Trans.}}^{l_i \rightarrow l} \right)$$

which models spread using transmitting influence, and,

$$\mathcal{M}_{\text{Share}}(m, L) = \underset{\{S \subseteq L \ | \ |S|=k\}}{\arg\max} \sum_{l \in S} \left(P_l^m + \sum_{l_i \in L-l} P_{l_i}^m \cdot \mathcal{I}_{\text{Share}}^{l_i \rightarrow l} \right)$$

which models spread using sharing influence.



(a) Transmitting Probability

(b) Sharing Probability

Figure 5.9: Clusters of related locations based on the transmitting and sharing probability functions.

To give a bit more insight into these two models, we constructed two directed graphs over the hashtag dataset – one graph for transmitting and other for sharing influence – with locations as nodes and the influence scores calculated using these functions as edge weights. In this graph, a cluster represents a collection of nodes (locations) that are close to each other, where closeness is defined either temporally (via transmitting influence) or based on content (via sharing influence). If the functions models location relationships correctly, then nodes that are close to each other in terms of distance should be in the same cluster (observation 1) and, nodes that are culturally similar should be the same cluster (observation 2). The results from this experiments are shown in Figure 5.9(a) and Figure 5.9(b), where every cluster is represented with a different color. In both these figures we can verify the two observations. Most locations which are close to each other are in the same cluster and some locations that are culturally similar, like the locations between English speaking parts of Western Europe and United States, and French speaking parts of Brazil and France, are in the same cluster.

5.4.2.3 Combining the Two Models

We can also combine the spatial and community affinity models by first defining an effective influence score:

$$Score(l) = P_l^m + \sum_{l_i \in L-l} P_{l_i}^m \cdot (\beta \cdot \mathcal{I}_{\text{Spatial}}^{l_i \to l} + (1-\beta) \cdot \mathcal{I}_{\text{Transmit}}^{l_i \to l}) \qquad (5.6)$$

where, β decides the weight assigned to each model and then using to model spread as:

$$\mathcal{M}_{\text{Spatial + Transmit.}}(m, L) = \underset{\{S \subseteq L \ | \ |S|=k\}}{\arg\max} \sum_{l \in S} Score(l)$$

We can define a similar model using sharing influence instead of transmitting influence as done above.

5.5 Experiments

In this section, we compare the quality of the proposed location selection approaches against three baseline approaches. We introduce metrics for measuring the quality of a selection approach, investigate the proposed approaches with respect to these quality metrics and identify the best approach to solve the location selection problem.

Approach	Accuracy	Impact	Impact Diff.
Random	0.256	0.343	0.739
Greedy	0.296	0.372	0.76
Lin. Regression	0.328	0.241	0.626
Sharing Infl.	0.266	0.264	0.666
Transmitting Infl.	0.242	0.253	0.654
Spatial Infl.	0.373	0.309	0.685
Transmitting Infl. + Spatial Infl.	0.407	0.393	0.78
Sharing Infl. + Spatial Infl.	**0.421**	**0.403**	**0.789**

Table 5.1: Comparing the predictive models ($t_s = 5$ minutes, $k = 3$). The approach combining the community influence approach with spatial influence approach (*sharing influence + spatial influence*) performs the best.

5.5.1 Baseline Approaches

In addition to the three geo-spatial approaches introduced in this section, we also consider three alternatives:

Random Selection: In this simplest approach, we randomly select k locations as the target subset, from the set of locations where the hashtag has occurred prior to t_s. The main drawback of this approach is that locations are selected without regard for the number of hashtags observed. In addition, since the target subset is selected based solely on a hashtag's propagation, the locations outside this set will never be selected. Hence, if the hashtag has occurred in fewer than k locations, then the target subset contains always fewer than k locations.

Greedy Selection: A natural improvement over random selection is a greedy approach, in which locations are selected based on the notion that a hashtag is going to continue to be used in locations where it is currently popular. Concretely, the greedy approach ranks locations based on the observed occurrence count of the hashtag: $|O_l^m|$. The intuition is that a hashtag that is popular in New York at location subset selection time is going to stay popular in the future as well. As in the random selection approach, it is possible that a hashtag might not have propagated to k locations, in which case we pick all the locations resulting in a subset with cardinality lesser than k.

Selection Based on Linear Regression: In this approach, we solve the location subset selection problem using a linear regression model. The idea behind this approach is to learn a model that can predict the unobserved occurrences for a hashtag given occurrences observed until the location subset selection time. Let M be the training hashtag set described in Section 5.3.2. Using M we first define the matrix X for observed occurrences as shown below:

$$X_i = \begin{pmatrix} 1 & \frac{|O_1^i|}{|O^i|} & \frac{|O_2^i|}{|O^i|} & \cdots & \frac{|O_{|L|}^i|}{|O^i|} \end{pmatrix} \quad \forall i \in [1, |M|]$$
$$X = \begin{pmatrix} \frac{|O_j^i|}{|O^i|} \end{pmatrix}_{|M| \times 1+|L|} = \begin{pmatrix} X_1 & X_2 & \cdots & X_{|M|} \end{pmatrix}^T$$

where, each row in this matrix corresponds to a hashtag in the training hashtag set. Similar to X, we define the unobserved matrix Y using unobserved occurrences.

$$Y_j = \begin{pmatrix} \frac{|U_j^1|}{|U^1|} & \frac{|U_j^2|}{|U^2|} & \cdots & \frac{|U_j^{|M|}|}{|U^{|M|}|} \end{pmatrix}^T \quad \forall j \in [1, |M|]$$
$$Y = \begin{pmatrix} \frac{|U_j^i|}{|U^i|} \end{pmatrix}_{|M| \times |L|} = \begin{pmatrix} Y_1 & Y_2 & \cdots & Y_{|L|} \end{pmatrix}$$

Using these matrices, we define Y as a linear function of X, $Y = X\beta + \mathcal{E}$, where, β is the $(|L| \times |L|)$ parameters matrix and $\mathcal{E}$ is the $(|L| \times |M|)$ matrix of error terms. Every column, β_l, in β models the relationships of a location l with the rest of locations and can be estimated by linear regression using the equation, $Y_l = X\beta_l + \mathcal{E}_l$, where $\mathcal{E}_l$ is the error column for l, in $\mathcal{E}$. We for a new hashtag m we can determine the top-k locations using:

$$\mathcal{M}_{\text{Lin. Reg.}}(m, L) = \underset{\{S \subseteq L \ | \ |S|=k\}}{\arg\max} \sum_{l \in S} \left(\hat{\beta_{l0}} + \sum_{i=1}^{|L|} \hat{\beta}_{li} \frac{|O_i^m|}{|O^m|} \right)$$

where, the expression in the parenthesis estimates probable occurrence distribution in locations for m.

5.5.2 Evaluation Metrics

We denote the best possible location subset that can be selected at t_s as $S_{t_s}^{m\star}$ ($S_{t_s}^m$ with a $\star$ on top). To evaluate the performance of the approaches proposed in this section, we define three metrics:

Accuracy: This metric measures the similarity between the approximate subset, determined using our approaches, and the exact location subset that is determined after the completion of hashtag propagation. This measure is similar to other set comparison metrics like the Jaccard index. It is defined as:

$$Accuracy = \frac{S_{t_s}^{m\star} \cap S_{t_s}^m}{k}$$

where, k is cardinality of $S_{t_s}^m$. If the sets are identical, the accuracy is 1.0, and 0.0 if they are disjoint.

Impact: While accuracy measures the similarity between the sets, it doesn't measure the effect of selecting a particular subset over another. For example, it is possible that two disjoint sets of locations observe same number of occurrences after they are selected, resulting in the same impact. Hence, we also consider the subset **impact**, which measures the percentage of hashtag occurrences that were observed in the approximate location subset. It is defined as:

$$Impact = \frac{U^m_{S^m_{t_s}}}{|O^m \cup U^m|}$$

where, the numerator is the number of occurrences that were observed in $S^m_{t_s}$, after it was selected, and the denominator is the total number of occurrences of the hashtag. The impact value ranges from 0.0 to 1.0, with 0.0 signifying no impact, while 1.0 signifying maximum impact.

Impact Difference: If a hashtag is distributed uniformly across large number of locations, then the best impact for a given k might be small. In this case, the performance of an approach will be measured as low, even if it selects the best set. Hence, we can also measure the subset **impact difference** that measures the difference between the impact for the best subset and the approximate subset. It is defined as:

$$Impact\ Difference = 1 - \frac{U^m_{S^{m\star}_{t_s}} - U^m_{S^m_{t_s}}}{|O^m \cup U^m|}$$

Like the other two metrics, the lower the value of difference the better is the approach. A value of 1.0 signifies the impact is identical while a value of 0.0 indicates the subset has no impact at all.

5.5.3 Evaluating the Models

We now evaluate the performance of location subset selection approaches using the metrics defined in the previous section. We first evaluate the performance of the approaches for a fixed value of location selection time t_s and subset cardinality k. We then evaluate the performance of these approaches by varying the time used to select location subsets. Similarly, we then evaluate the performance of the approaches for different sizes of location subsets.

Experimental Setup: For our experiments we use two hashtag sets: (i) Training hashtag set, and (ii) Test hashtag set. The hashtag sets are extracted from Twitter hashtag propagations as described in Section 5.3.2. Techniques that require prior hashtag propagations (linear regression, sharing and transmitting influence) use the training hashtag set to build their models. For the spatial influence model, we set $\alpha = 1.01$.

We use the test hashtag set to evaluate the performance of the approaches. Given a hashtag from the test set, to evaluate an approach-metric pair, we replay the hashtag's propagation. At location subset selection time, we select location subset using this approach and then continue with the remaining propagation of the hashtag. At the end of this hashtag's propagation, we measure performance of the approach using this particular metric. We do this for all hashtags in the test set and calculate the mean score for this metric-approach pair. This experiment is done for a given value of t_s and k. We set $\beta = 0.5$ in (5.6) giving equal weight to both approaches.

Comparing the Models: We begin by fixing the selection time for each approach as 5 minutes (i.e., $t_s = 5$) and the number of locations to selects as 3 (i.e., $k = 3$). How well do the approaches predict future locations given only evidence of the first 5 minute's of a hashtag's lifetime? We report the results across all approaches for accuracy, impact, and impact difference in Table 5.1. Recall that accuracy measures the similarity between subsets selected by our approaches and the best subset; impact measures the actual percentage of occurrences observed in the locations; and impact difference measures the percentage difference between the best impact and the impact achieved using one of the approaches.

First, we observe that the approach combining the community influence approach

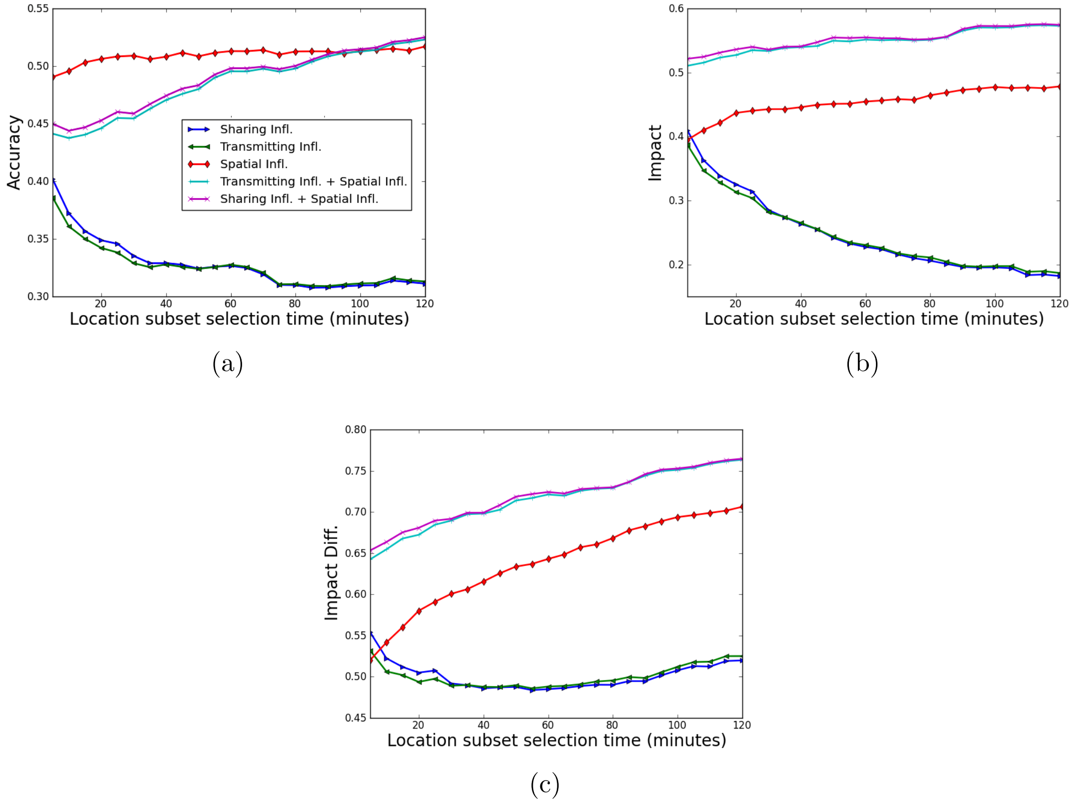
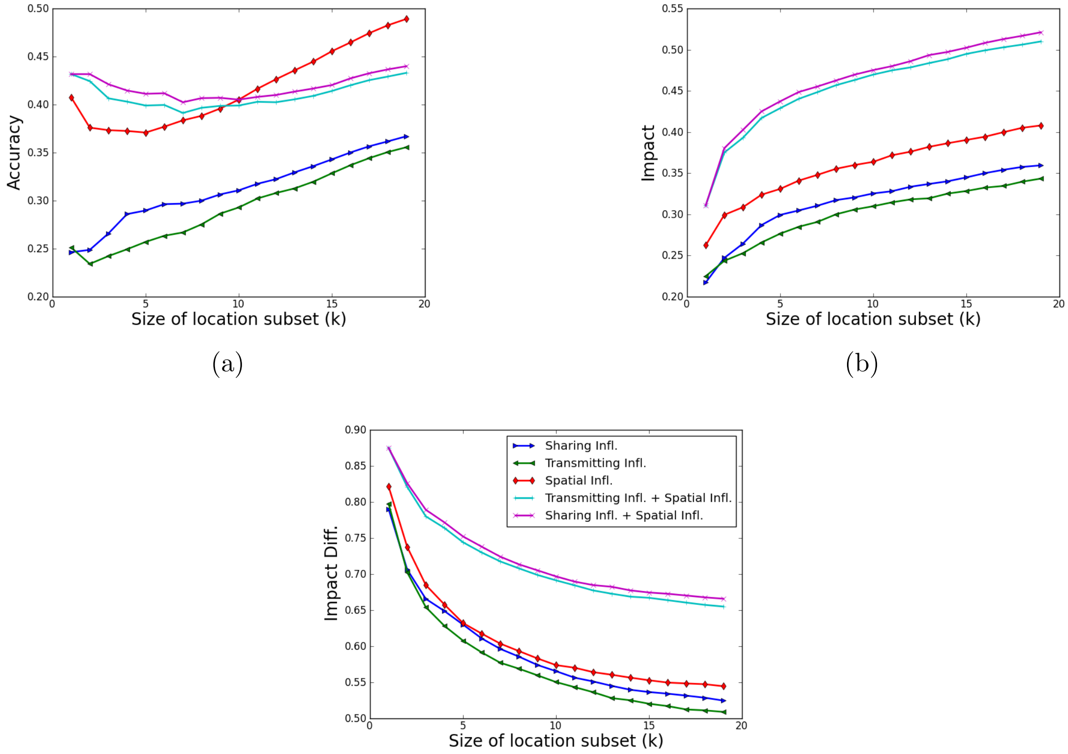


Figure 5.10: Result of modeling after varying the selection time.

with spatial influence approach (*sharing* + *spatial*) performs the best, with an accuracy of 42%, and impact of 40%, and an impact difference of 79%. Interestingly, we observe that approaches based on the spatial influence model tend to perform much better than approaches that use only historical hashtag propagations (e.g., linear regression). For example, the accuracy of the *spatial influence*, of *transmitting* + *spatial*, and of *sharing* + *spatial* is higher in all cases than all other approaches. We see similar strong results for the combined approaches (*transmitting* + *spatial*, and of *sharing* + *spatial*) as compared to all other approaches. Surprisingly, the community influence-based approaches alone (e.g., *sharing* and *transmitting*) perform the worst, even worse than the random and greedy approaches.



(a)

(b)

(c)

Figure 5.11: Result of modeling after varying the number of locations predicted.

These results are significant because they illustrate the importance of prioritizing the spatial influence model over the community affinity models, but also the combined power of incorporating community affinity into the spatial influence model for the best overall performance. Selecting future locations that will adopt a hashtags with very little knowledge of how a hashtag is going to propagate is a difficult problem. Based on these results, the performance achieved by the model that combines sharing probability with coverage probability is very encouraging. Most popular hashtags spread for several hours, but this model can identify 40% of all future occurrences of a hashtag within 5 minutes of the hashtag's first appearance. Also, the quality of locations selected by this model is high, as the locations it selected came close to

79% of the best performing locations.

Varying the Selection Time: What if we increase the time until the models have to make a prediction? That is, if we allow the hashtags to propagate for even longer, what impact does this have on the predictive ability of the models as they have access to additional evidence? Hence, we next varied the location subset selection time (t_s) from 5 minutes to 2 hours, keeping the k fixed at 20. We evaluated each approach for each selection time (e.g., after 5 minutes since a hashtag's first appearance, after 10 minutes, and so on up to 120 minutes) as shown in the Figure 5.10. We plot the affect of varying the selection time against the five approaches, showing accuracy in Figure 5.10(a), impact in Figure 5.10(b), and impact difference in Figure 5.10(c).

We see that across all metrics, the approaches that use both sharing and transmitting influence coupled with spatial influence (the purple and light blue curves) improve with the increase in location selection time. As the time to select locations increases, each approach can observe a longer lifespan of a hashtag's propagation, leading to stronger evidence for making better predictions. In contrast, the community affinity approaches alone (*sharing* and *transmitting*, in blue and green) degrade in quality as the selection time increases (with a slight uptick for impact difference after 80 minutes). These results further confirm the importance of the spatial influence models as the single strongest predictor of hashtag spread.

An interesting result we observe in this figure is the performance of approach that uses spatial influence alone to select locations. We observe that the curve (red-diamonds) corresponding to this approach stays relatively constant irrespective of the value of t_s. This approach selects locations just based on spatial influence and hashtag distribution, hence a constant accuracy indicates that the probability scores for locations remain same irrespective of t_s, i.e., the overall probability distribution for a hashtag calculated after 5 minutes is similar to its probability distribution calculated after 2 hours. This result further strengthens our assessment, in Section 5.3.3, that early coverage for a hashtag is a good indicator of its final coverage.

Confirming the results from our previous experiment, we find that approaches that use the spatial influence model in concert with a community affinity model perform the best.

Varying the Number of Predictions: Finally, we evaluate the performance of each approach by varying the number of locations each predicts. Hence, we vary the cardinality k from 1 to 20, while fixing the selection time at 5 minutes, as shown in the Figure 5.11. Across all three metrics – accuracy in Figure 5.11(a), impact in Figure 5.11(b), impact difference in Figure 5.11(c) – we again see the strong performance of the spatial influence models, both for the spatial model along (*spatial*) as well as the model incorporating community affinity into the spatial model (*transmitting + spatial* and *sharing + spatial*). As the number of locations increases, we see the accuracy of all approaches increase since each selects more top locations correctly. We also see an improvement in impact for all the approaches, with increasing cardinality. This result is straightforward since increasing the number of locations implies a higher number of occurrences are observed, which in turn increases the impact. But, the magnitude and rate for improvement of impact varies for all the approaches, with all the approaches that use spatial influence model showing greater impact than approaches that use community affinities only. This result is similar to the results observed in Figure 5.10(b). Finally, we observe that increasing the cardinality results in a decrease in impact difference for all approaches.

5.5.4 Summary of Results

Based on this experimental study, we find that:

- First, **distance does matter**. As shown in Table 5.1, we found that the spatial influence model – based on Tobler's first law of geography – is the single most important explanation of future hashtag adoption. Distance matters mostly because hashtags are fundamentally a *local phenomena*. Hashtags typically occur in an originating location and subsequently in nearby neighboring locations.

- Second, we additionally discovered that though the community affinity influence model alone performs worse than the spatial influence mode, **in combination** with the spatial influence model we can achieve the best fit for future hashtag adoption. This combination indicates that community affinities (like culture, language, and common interests) are a secondary factor

5.6 Summary

In this section, we have begun an investigation of the global spread of social media. We have studied the geo-spatial properties of a collection of 755 million geo-tagged tweets and found that (i) pairs of locations tend to share common hashtags and adopt them around the same time, compared with locations that are distant; (ii) many distant location pairs, though, exhibit similar patterns of "closeness" in that they share hashtags and have a low hashtag adoption lag, suggesting some additional factor is "bending space" to link the two locations; and (iii) the early occurrences of a hashtag are good indicators of the relative coverage of a hashtag's future spread (either compact or widely diffuse). Based on these observations, we developed two complementary models of hashtag spread – the spatial influence model and the community affinity influence models – and studied their effectiveness at predicting locations that will adopt hashtags in the future. We conclude that **distance does matter** as the single most important explanation of future hashtag adoption since hashtags are fundamentally local. We also find that community affinities (like culture, language, and common interests) enhance the quality of purely spatial models, indicating the necessity of adequately incorporating non-spatial features into models of global social media spread.

In our continuing work, we are interested in augmenting the developed models – that consider only the geo-spatial properties of hashtags – with additional evidence of the content of the hashtags (e.g., since politics-related social media may spread differently than sports-related social media) and with the underlying social network. Recall that the study in this section has been completely orthogonal to the underlying social network and how social contagion affects hashtags spread. As part of this continuing work, we are interested in linking these geospatial diffusion models to these related efforts.

6. REAL-TIME RECOMMENDATION OF SOCIAL TRAILS

In this section, we begin our investigation of social trails analytics. In particular, we focus on developing methods that can be used to predict future occurrences of social trails and hence can be used in real-time social trail recommendation. As in the previous section, we use hashtag propagation as instance of social trails in our analysis and experiments. Our proposed methods model the geo-spatial propagation of online information spread to identify which hashtags will become popular in specific locations. Concretely, we develop a novel reinforcement learning approach that incrementally updates the best geo-spatial model. In experiments, we find that the proposed method outperforms alternative linear regression based methods.

6.1 Introduction

The widespread adoption of GPS-enabled tagging of social media content provides new access to the fine-grained spatio-temporal logs of user activities. For example, the Foursquare location sharing service has enabled 2 billion "check-ins" [28], whereby users can link their presence, notes, and photographs to a particular venue. The mobile image sharing service Instagram allows users to selectively attach their latitude-longitude coordinates to each photograph; similar geo-tagged image sharing services are provided by Flickr and a host of other services. And the popular Twitter service sees 500 million Tweets per day, of which around 5 million are tagged with latitude-longitude coordinates.

With access to the worldwide geo-spatial footprints of social media users, we focus on the problem of *predicting what online memes will be popular in what locations*, which has important implications for a variety of systems and applications, including targeted advertising, location-based services, social media search, and content delivery networks. In particular, we focus our investigation on a sample of 755 million geo-tagged Tweets with precise latitude-longitude coordinates collected over the course of 18 months. Specifically we consider the propagation of hashtags across Twitter, where a hashtag is a simple user-generated annotation prefixed with a #. Hashtags serve many purposes on Twitter, from associating Tweets with particular events (e.g., #ripstevejobs and #fukushima) to sharing memes and conversations (e.g., #bestsportsrivalry and #ifyouknowmeyouknow).

Our goal is to develop techniques based on Twitter hashtag propagation which can be used to predict hashtags that will be popular at any location. For example, can we accurately predict which hashtags will be popular in San Francisco over the next two hours? Can the same model also predict which hashtags will be popular in a small town like College Station, Texas? Can we identify which hashtags that have been popular in New York in the past two hours but will drop in interest? Building robust models that can accurately predict the spatio-temporal popularity of online memes like hashtags can aid in design of systems and applications, including content delivery networks, social media search, location-based services like Google Now, and geo-targeted advertising.

Toward answering these questions, we develop in this section a reinforcement learning-based approach that builds upon two competing hypotheses of information spread over geo-spatial networks.

- **Spatial Affinity:** The first hypothesis, based on the Tobler's first law of geography [72], states that the information spread between two locations is impacted by the distance between two locations. For example, according to this hypothesis hashtags spread faster between San Francisco and Mountain View, since they are closer to each other; but slower between San Francisco and Austin.

- **Community Affinity:** The second hypothesis is that the "world is flat" and information spreads based on virtual communities enabled by the prevalence of the Internet. In this hypothesis, distance is less important than are the strength of these virtual ties between locations; e.g., under this hypothesis San Francisco and Austin may be considered closer in terms of common interest (and hence, hashtags should flow more rapidly between the two), rather than Austin and its more proximate neighbor Houston.

We investigate a series of features inspired by these two hypotheses for predicting which hashtags will be popular in a specific location at a specific time. Since the best features may vary for each location, we additionally propose a reinforcement-learning based method whereby the best model is determined is location specific. In our experimental evaluation of over 755 million geo-tagged Tweets, we find that reinforcement learning algorithm that selects the single best feature function for a location performs the best. This model is able to predict close to 70% of future hashtags occurrences accurately.

6.2 Related Work

The area of information diffusion is well studied with most work focussed on study of diffusion through social and information networks, e.g., [32, 42, 43, 44, 46, 78]. But, our work in particular builds on two lines of research: Twitter analysis and geo-spatial analysis of social media.

Twitter Analysis: Most papers studying Twitter have focused on understudying its properties as a social network and have tried to analyze information diffusion as a effect of the underlying social network [37, 45, 46, 78]. Hence, similar models have been applied to study hashtag propagation on Twitter's social network [64, 17]. In related research, people have studied approaches to predict the popularity of hashtags in a given time frame in [73], sentiment detection on Twitter [20], topic tracking on Twitter streams [48], and so forth.

Geo-spatial Analysis of Social Media: In recent years we have seen large-scale geo-spatial analysis motivated by the emergence of location-based social networks like Foursquare, Gowalla, Google Latitude, and so on [67, 56, 4, 5, 22, 39]. A recent paper dealt with the spatial analysis of YouTube videos [9]. In this work the authors were able to observe the highly local nature of videos based on the propagation patterns of YouTube videos. On Twitter, geo-spatial analysis has focused on inferring geographic information from tweets like predicting user locations from tweets [12] and spatial modeling to geolocate objects [19].

6.3 Twitter Data Collection

We collected a sample of around 755 million geo-tagged tweets containing around 10 million unique hashtags from Twitter using the Twitter Streaming API from February 1 to November 30, 2011. Each tweet in this sample is tagged with a latitude and longitude indicating the location of the user at the time of the posting. Each <`hashtag`, `time`, `latitude`, `longitude`> tuple correspond to a particular hashtag occurrence.

To support location-based analysis, we divide the globe into square grids of equal area using Universal Transverse Mercator (UTM), a geographic coordinate system which uses a 2-dimensional Cartesian coordinate system to map locations on the surface of the globe [75]. The issue with using an angular co-ordinate system like latitudes and longitudes is that distance covered by a degree of longitude differs as we move towards the pole. In addition, the distance covered by moving a degree

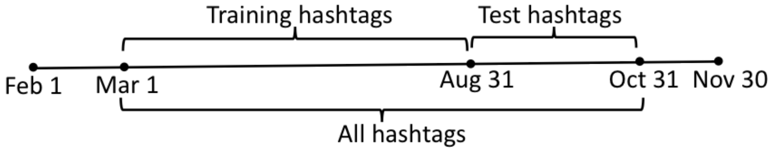


Figure 6.1: Hashtags datasets.

in latitude and longitude is same only at the equator. Hence, it is hard to break the globe into grids using this system. UTM on the other hand gives us a system of grids that closely matches distances in metric system making our analysis easier. While varying the choice of grid size can allow analysis at multiple levels (e.g., from state-sized cells to neighborhood-sized ones), we adopt a middle ground by dividing the globe into squares of 10km by 10km. Some grid cells will naturally be densely populated, others will be sparse. Let this set of distinct locations, each corresponding to a square, be represented by the set L.

To avoid sparsely represented hashtags, we consider only hashtags with at least 5 occurrences in a location and consider only hashtags with at least 250 total occurrences across all locations. Since some hashtags may have begun their Twitter life before the first day of our sample (February 1) while others may have continued on after the last day (November 30), we consider both February and November as buffer months. Hence, we capture the full lifecycle of hashtags starting on or after March 1 and ending by October 31, which focuses our study to hashtags which have both their birth and death within the time of study (and as a result, removes cyclical hashtags like "#ff" and "#nofollow"). As illustrated in Figure 6.1, we additionally divide the set of all hashtags into two sets: a training set based on hashtags from March to August; and a test set based on September to October. Hashtags that start in training but continue into test are ignored. In this way, the training set contains 1466 complete hashtag propagations and the test set contains 515.

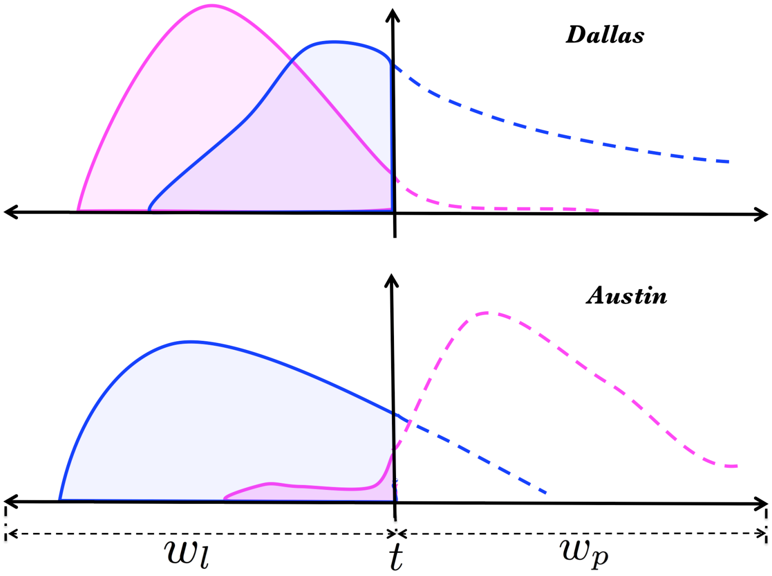


Figure 6.2: Example of trail propagation in two locations

6.4 Spatio-Temporal Meme Prediction

Let H be a set of hashtags and L the set of distinct locations. Then for a hashtag $h \in H$ let o_l^h be the number of occurrences of the hashtag that have been observed in a location $l \in L$, and let e_l^h be the number of occurrences of the hashtag that are expected in l. We now define the problem of selecting top$-k$ hashtags for a location as **hashtag subset selection problem**.

Definition 6.4.1. ***(Hashtag Subset Selection Problem)****: Given an integer k, hashtag subset selection problem for a location l is the task of determining set of top$-k$ hashtags $S_l \subseteq R$ such that the total number of expected hashtags for S_l is maximized, i.e.,*

$$S_l = \underset{\{S \subseteq R \ \mid \ |S|=k\}}{\arg\max} \sum_{h \in S} e_l^h \tag{6.1}$$

To understand the hashtag subset selection problem better, consider the example shown in Figure 6.2. It shows propagation of two hashtags (pink and blue) in Dallas and Austin at time t. The number of observed and unobserved occurrences for these hashtags at a time t is indicated by the area below shaded region with solid lines and a unshaded region with dotted lines respectively. Now, given that we only know the shaded regions under complete lines at t, the hashtag subset selection problem is the task of identifying k hashtags that will have maximum area under dotted lines. If $k = 1$, the solution to this problem would be $S_{Dallas} = \{\text{Blue}\}$ and $S_{Austin} = \{\text{Pink}\}$.

Feature Functions: If we know the area under dotted lines, i.e., e_l^h, then the solution to this problem is trivial. But, since we don't have that information at t, we have to develop methods to estimate this value. Let $\hat{e}_l^h$ be a score representing the value of e_l^h. Depending upon the method used to estimate this score, it could be anything – an integer predicting the number of expected occurrences or a value $\in [0, 1]$. The only condition is that a higher score for a location should indicate that this location sees more occurrences than a location with a lower score. Then using (6.1), we redefine the hashtag subset S_l in terms of $\hat{e}_l^h$ as:

$$S_l = \underset{\{S \subseteq R \ | \ |S|=k\}}{\arg\max} \sum_{h \in S} \hat{e}_l^h \tag{6.2}$$

As mentioned earlier, the score, $\hat{e}_l^h$, for a location l and hashtag h, can be determined using several techniques. Let F be the set of feature functions used to estimate the value of e_l^h, where, $f_i \in F$ is defined as $f_i : L \times H \to \mathbb{R}$. For example, a simple way to estimate expected hashtags in a location would be to use the notion that a hashtag that is popular in that location at the current time will continue to be popular there during the future. This would say that a hashtag (#redskins) about a football game in Washington D.C that is popular right now can be expected to remain popular next hour too. Concretely, calling this the greedy approach we can define the feature function corresponding to this, $f_{\text{greedy}} \in F$, as:

$$f_{\text{greedy}}(l, h) = o_l^h$$

where f_{greedy} just gives the number of occurrences of h that have been observed in l.

Learning Algorithms: Every feature function in F estimates a different value of $\hat{e}_l^h$, i.e., for a given location-hashtag pair we have $|F|$ estimates for $\hat{e}_l^h$. But, for a given location-hashtag pair, we can only use one value of $\hat{e}_l^h$ in (6.2). So, we formulate the task of determining a single value from a set of $|F|$ values as a learning problem. In particular, we propose a set of learning algorithms, $\mathcal{L}$, that use the set of feature functions F and a location-hashtag pair to estimate the value for $\hat{e}_l^h$. The learning algorithm can either combine all the estimated values in some ratio to get a new value of $\hat{e}_l^h$ or use some heuristic and select one of the values that it thinks is the best estimate. For example, we can estimate a new value for $\hat{e}_l^h$ using linear regression as:

$$\hat{e}_l^h = \epsilon + \sum_{f_i \in F} w_{f_i} f_i(l, h)$$

where, w_{f_i} are regression coefficients and ϵ is the error term.

In the following two sections we address two fundamental questions:

- **Feature Functions**: What feature functions F do we use to determine the value of $\hat{e}_l^h$?
- **Learning Algorithms**: What learning algorithms do we use to determine a single value from a set of $|F|$ values of $\hat{e}_l^h$?

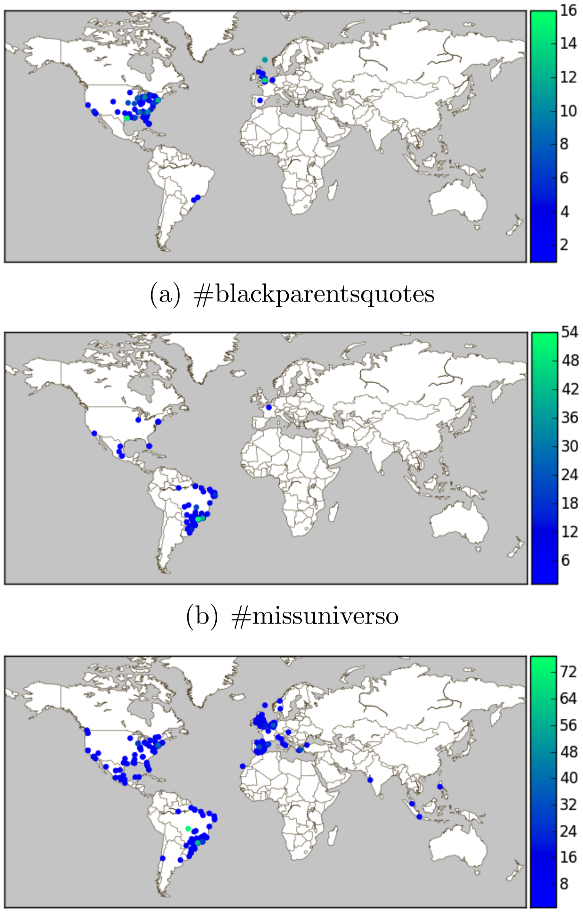
6.5 Feature Functions

The feature functions to estimate the expected number of hashtag occurrences are guided by two major concepts of geo-spatial propagation: spatial affinity and community affinity. We first describe the feature functions based on spatial affinities where one function estimates local hashtags more accurately while another estimates global hashtags more accurately. We then describe feature functions that use community affinities to learn relationship between locations.

6.5.1 Spatial Affinities

In this section, we present feature functions that use spatial affinities between locations as described by the Tobler's hypothesis [72] to estimate expected number of hashtags. Tobler's hypothesis implies that the popularity of a hashtag in a location is dependent on the popularity of this hashtag in neighboring locations. So, we predict the future popularity of a hashtag in a particular location as a function of the hashtag's spatial distribution in other locations, such that the "contribution" made by the other location decreases exponentially as its distance from the particular location increases.

An advantage of using spatial affinities to estimate expected hashtag occurrences is that this approach allows us to develop different feature functions depending on our preferred hashtag type – local hashtag or global hashtag. Examples of local and global hashtags are shown in Figure 6.3. It shows spatial distribution for two local hashtags - #blackparentsquotes (USA and England) and #missuniverso (Brazil), and one global hashtag - #usopen (entire world), which were popular on the evening of September 19, 2011. We can imagine applications (like localized advertising) where we would want to prefer one type of hashtag over other and the feature function based on spatial affinities helps in such cases. In particular, we propose two feature



(a) #blackparentsquotes

(b) #missuniverso

(c) #usopen

Figure 6.3: Distribution of three trails on the evening of September 19, 2011.

functions: (i) global feature function which is suitable to estimate hashtags that are globally popular; and, (ii) local feature function which is suitable to estimate local hashtags.

Global Feature Function: This function uses spatial distribution of hashtags and estimates global hashtags more accurately than local. It is similar to the greedy feature in the sense that both these functions use a hashtag's observed occurrences to estimate expected hashtag occurrences. But, unlike greedy, this approach doesn't use raw occurrence counts but *shifted occurrence counts.* Shifted occurrences are occurrences that are contributed to a location from other locations using Tobler's hypothesis, such that locations that are close by contribute greater number of occurrences to the location than locations that are far off. The global feature function is defined as:

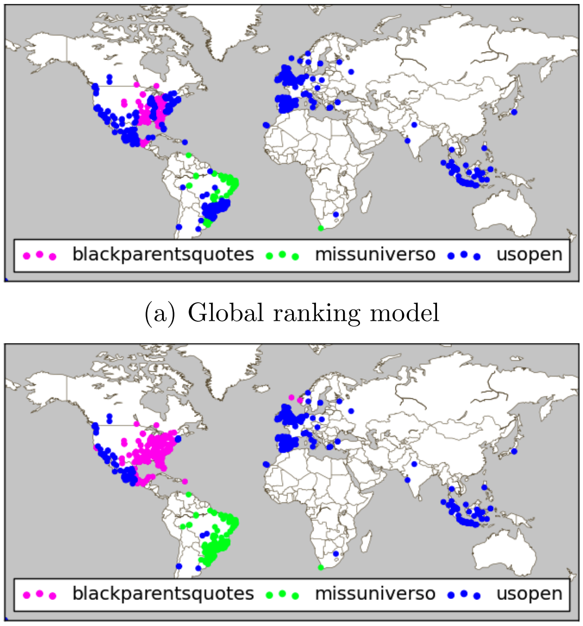
$$f_{\text{global}}(l,h) = \sum_{l_i \in L} o_{l_i}^{h} \alpha^{-H(l_i,l)}$$

where, the sum calculates the total number of shifted footprints of h contributed by all locations to l. The exponential function helps model Tobler's hypothesis by decaying the contribution made by l_i to l depending on the distance between the two locations. The parameter α controls the rate of decay and in our experiments we set $\alpha = 1.01$.

Local Feature Function: As mentioned earlier, this feature function uses spatial distribution to estimate expected hashtag occurrences for local hashtags more accurately than global hashtags. But, instead of estimating expected count this feature function estimates a score in $[0,1]$ that is an indicator of expected hashtag occurrences, such that, a higher score for a location indicates that more hashtags are expected at that location than a location with lower score. But, before describing this function, we first define the probability of observing a hashtag h in l, P_l^r, as:

$$P_l^h = \frac{o_l^h}{\bigcup_{l_i \in L} o_{l_i}^h}$$

The score is calculated by applying Tobler's hypothesis to the hashtag observing probability. So, we define the local feature function for a hashtag h in a location l



(a) Global ranking model

(b) Local ranking model

Figure 6.4: Ranking trails using geospatial distribution

as:

$$f_{\text{local}}(l,h) = \frac{\sum_{l_i \in L} P^h_{l_i} \alpha^{-H(l_i,l)}}{\sum_{l_j \in L} \sum_{l_i \in L} P^h_{l_i} \alpha^{-H(l_i,l_j)}}$$

where, the numerator sums the shifted hashtag occurrence probability values from all locations to l. The exponential term is used to model Tobler's hypothesis such that locations that are closer to l contribute more to the score than locations that are far from l. Like before, in our experiments we set $\alpha = 1.01$.

To illustrate differences between the two spatial affinity based feature functions described in this section, consider the spatial distributions of three hashtags shown in Figure 6.3. We use the global and location feature selection methods to predict the expected number of occurrences for each of these hashtags. Then we mark every location with the color of the hashtag that was most accurately estimated. The performance of these feature functions is shown in Figure 6.4. In these figures we observe the difference in approaches that the two feature functions take to estimate the expected number of occurrences for local and global hashtags. The global feature function as expected estimates hashtag occurrences for global hashtags more accurately as shown by the blue locations in Brazil and USA where other local hashtags exist. The local feature function on the other hand, estimates the excepted occurrences of local hashtags #blackparentquotes (pink) and #missuniverso (green) more accurately.

6.5.2 Community Affinities

The approaches proposed so far took into account only the geographical distances between two locations to estimate expected hashtag occurrences. In this section, we move beyond geographical distances and look at an alternative approach that considers the impact of virtual communities that exist over the Internet. In particular, we present feature functions that use community affinities between locations that may not necessarily be close in terms of geographical distance. In particular, we propose two feature functions that differ in the way community affinities between two locations is measured: (i) common hashtags feature function uses community affinities measured based on the set of common hashtags shared between locations; and, (ii) hashtag transmission feature function uses community affinities between locations measured based on the hashtags that a location might have transmitted to another. Both these approaches learn affinities between locations based on historical hashtag propagations. To do this we use the training set described in Section 6.3. Let H^T be the set of all hashtags observed in the training set and $H_l^T \in R$ the set of hashtags observed in location l. Then, we define a prior probability of observing a hashtag in

l as:

$$P_l^T = \frac{|H_l^T|}{|H^T|}$$

We define $\mathcal{C}_{l_i \to l_j} \in [0, 1]$ as the measure of community affinity between locations l_i and l_j such that, $\mathcal{C}_{l_i \to l_j} = 1.0$ indicates that a hashtag in l_i will definitely occur in l_j and $\mathcal{C}_{l_i \to l_j} = 0.0$ indicates that a hashtag in l_i will not occur in l_j.

Common Hashtags Feature Function: In this approach we measure community affinities between locations based on the information about common hashtags observed between a pair of locations. The intuition behind this approach is that if locations are connected by virtual communities then they must share common hashtags. Ex: techies in San Francisco and techies in Austin though geographically apart will share common hashtags. For the pair of locations l_i and l_j we define the common hashtag affinity $\mathcal{C}^{com}_{l_i \to l_j}$ when a hashtag has occurred in l_i, as:

$$\mathcal{C}^{com}_{l_i \to l_j} = \frac{|H^T_{l_i} \cap H^T_{l_j}|}{|H^T_{l_i}|}$$

Note that there might be cases where $\mathcal{C}^{com}_{l_i \to l_j} \neq \mathcal{C}^{com}_{l_j \to l_i}$ as the number of hashtags observed in these locations might be different $(|R_{l_i}| \neq |R_{l_j}|)$. We now define common hashtag feature function using affinities learned from common hashtags observed in locations as:

$$f_{\text{com}}(l, h) = \sum_{l_i \in L - l} P^T_{l_i} \; P^h_{l_i} \; \mathcal{C}^{com}_{l_i \to l_j}$$

where, the sum calculates the total influence other locations have on l to make a hashtag h popular.

Hashtag Transmission Feature Function: After looking at affinities based on common hashtags observed in locations, we now look at affinities based on a set of hashtags that a location might have transmitted to another. In particular, with this approach we are interested in learning affinities that can reflect temporal relationships between locations. We define the affinity, $\mathcal{C}^{tran}_{l_i \to l_j}$, measured this way as hashtag transmission affinity and it indicates the chance that a hashtag observed in a particular location will be observed in another location in the future. For example, in Figure 6.2, observing that the pink hashtag that is popular in Dallas during the estimation window becomes popular in Austin during the prediction window, we can learn the temporal relationship between these two locations. We define $\mathcal{C}^{tran}_{l_i \to l_j}$, as:

$$\mathcal{C}^{tran}_{l_i \to l_j} = \frac{|\{h \mid t^h_{l_j} > t^h_{l_i} \quad \forall h \in H^T_{l_i} \cap H^T_{l_j}\}|}{|H^T_{l_i}|}$$

where, t^h_l is the location l's traction time for h. The numerator in this definition is the size of set of hashtags that gained traction in l_i before l_j. Like in case of affinities based on common hashtags, there might be cases where $\mathcal{C}^{tran}_{l_i \to l_j} \neq \mathcal{C}^{tran}_{l_j \to l_i}$. Similar to the common hashtag feature function, the hashtag transmission feature function is defined as:

$$f_{\text{tran}}(l, h) = \sum_{l_i \in L - l} P^T_{l_i} \; P^h_{l_i} \; \mathcal{C}^{tran}_{l_i \to l_j}$$

An example of how community affinities differ from spatial affinities is shown in Table 6.1 and Table 6.2. In this table we compare the community (common hashtag) and spatial affinities for Austin, Texas. We observe that though Austin is spatially closer to some of the other big cities in Texas, the hashtags observed there are more similar to the hashtags observed in Los Angeles, Washington D.C and New York.

City	Distance (miles)	Affinity
San Antonio	79	0.54
Houston	167	0.79
Dallas	191	0.92

Table 6.1: Example of spatial affinities for Austin

City	Distance (miles)	Affinity
Los Angeles	1,373	0.96
Washington D.C	1,520	0.94
New York	1,732	0.92

Table 6.2: Example of common hashtag affinities for Austin

6.6 Learning feature functions

In the previous sections we proposed five feature functions to estimate $\hat{e}_l^h$. First the greedy feature function and then two feature functions that used the hypothesis that distance between two locations played an important role in making hashtags popular. Finally two more feature functions that used a contradictory hypothesis that it wasn't distance but virtual communities on Internet that impact hashtag popularities.

The next task is to reduce $|F|$ values of $\hat{e}_l^h$ to a single value that can be used in (6.2). A simple approach now would be to evaluate which of these feature functions determines the value of $\hat{e}_l^h$ most accurately and select it. In reality though, we might observe that a single feature function might not be suitable for all locations. Instead the demography of a place might dictate selection of a particular function that is best for this place. For example, metropolitan areas like those around Austin might prefer community feature functions, while smaller towns surrounding Dallas might prefer the spatial feature functions. In addition, it is possible that some locations might not prefer one feature function over the other but a combination of these feature functions in some ratio. Hence, in this section to deal with these issues we concentrate on two things: (i) introduce evaluation metrics to measure the performance of feature functions for a location; and, (ii) describe algorithms that use these metrics to learn the best feature function or the best ratio for combining the feature functions for a location.

6.6.1 Evaluation Metrics

We now describe two evaluation metrics which we use in learning the best feature function or best combination of feature function for a given location. The value for each of these metrics is in the range $[0, 1]$ with 0.0 indicating the worst performance and 1.0 indicating the best performance. Given a location l, we denote the best set of top$-k$ hashtags at this location as ${S_l}^\star$ and the set of top$-k$ hashtags selected by our ranking models as S_l (without the $\star$ on top). The two evaluation metrics are:

Accuracy: This metric measures the similarity between ${S_l}^\star$ and S_l. This measure is similar to other set comparison metrics like the Jaccard index. It is defined as:

$$\mathcal{A}_l = \frac{{S_l}^\star \cap S_l}{k}$$

such that, if the sets are identical accuracy is 1.0 and 0.0 if they are disjoint.

Impact: While accuracy measures the similarity between the sets, it doesn't measure the effect of selecting a particular set of hashtags over another. For example, it is possible that two disjoint sets of hashtags might observe same number of total hashtag occurrences after they are selected, resulting in the same performance. Hence, we define a metric called hashtag subset's impact defined as:

$$\mathcal{I}_l = \frac{\sum_{h \in S_l} e_l^h}{\sum_{h \in S_l{}^\star} e_l^h}$$

which measures the ratio between the number of hashtag occurrences that were observed for hashtags in S_l to those in $S_l{}^\star$. The impact value 0.0 signifies no impact while 1.0 signifies best impact.

6.6.2 Learning Algorithms

We next describe learning algorithms to determine a single value for $\hat{e}_l^h$ from $|F|$ values for it estimated using the feature functions. We build a different model for each location $l \in L$ and to build these models we use the training and test hashtag sets described in Section 6.3, which contains complete propagations for all hashtags. In particular, we present two learning algorithms depending on how the learning algorithm assigns best feature function to a location: (i) linear regression algorithm which determines the weights for a linear combination of feature functions for a location; and, (ii) reinforcement learning algorithm which determines the single best feature function for a location.

Learning with Regression: We first describe a learning algorithm using linear regression to determine a single value for $\hat{e}_l^h$, where a different model is built for each location $l \in L$. To build these models we use the training and test hashtag sets described in Section 6.3. We know the complete propagation for a hashtag in the training and test sets. Consider the matrices X_l and Y_l:

$$X_l = \begin{pmatrix} 1 & f_1(l,h_1) & f_2(l,h_2) & \cdots & f_{|F|}(l,h_{|H|}) \\ 1 & f_1(l,h_1) & f_2(l,h_2) & \cdots & f_{|F|}(l,h_{|H|}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & f_1(l,h_1) & f_2(l,h_2) & \cdots & f_{|F|}(l,h_{|H|}) \end{pmatrix}$$

$$Y_l = \begin{pmatrix} e_l^{h_1} & e_l^{h_2} & \cdots & e_l^{h_{|H|}} \end{pmatrix}^T$$

where, X_l matrix has $|H|$ rows, one for each hashtag in the training set. Every row contains $1 + |F|$ values each, except that for the first column, corresponding to the expected value for the hashtag calculated using the feature function corresponding to the column. The column matrix Y_l has $|H|$ rows with each value equal to the real expected value determined from the training set.

The values for the matrices is calculated using learning (w_l) and prediction (w_p) windows as shown in Figure 6.2. Note that, the expected value in Y_l increases as we increase the prediction window, i.e., using a prediction window of 4 hours will have more hashtag occurrences than a window of 2 hours. Similarly, the observed hashtag occurrences used by feature functions to determine X_l varies as the learning window is varied. The impact of varying these windows on the learning algorithms is evaluated later in the experiment section. Using these matrices, we define Y_l as a linear function of X_l,

$$Y_l = X_l \beta_l + \mathcal{E}_l \tag{6.3}$$

where, β_l is a column matrix called parameters matrix and $\mathcal{E}$ is the matrix of error terms. The parameters matrix contains the weights using which the various feature functions should be combined to determine $\hat{e}_l^h$ from $|F|$ estimates for it. The parameters matrix can be estimated by linear regression using the equation (6.3). We for a new hashtag h we can determine the expected occurrences for it using:

$$\hat{e}_l^h = \hat{\beta}_0 + \sum_{i=1}^{|F|} \hat{\beta}_i f_i(l, h)$$

Learning with Reinforcement: In the previous method we used linear regression to combine the values of expected hashtag occurrences estimated by all the feature functions. We now describe an approach that uses reinforcement learning to determine this value. By reinforcement learning we mean that during every time interval the learning algorithm makes some prediction, then in the next time interval it updates its model based on its performance before making future predictions.

The learning algorithm is run independently for every location at regular time intervals. Let the weight $W_l^f(t)$, for every location-feature function pair, represent the value that the learning algorithm uses to select a feature function for a given location at time t. During every time interval we select a feature function that we expect will perform best using $W_l^f(t)$. We then evaluate the performance of all of all the feature functions using some metric (accuracy or impact) and update the $W_l^f(t)$ accordingly. So, the idea is that after a few observations the algorithm learns which feature function is best suited for a location.

We describe two methods of reinforcement learning depending upon how $W_l^f(t)$ is updated and used to select a feature function: (i) Deterministic method which selects the best feature function at any time; (ii) Randomized method which uses a probability to select a feature function.

Deterministic Method: This method selects the single best feature function for a location. In this method the weight $W_l^f(t)$ for every location-feature function represents the cumulative loss for the function until time t:

$$W_l^f(t) = W_l^f(t-1) + (1 - \mathcal{A}_l^f)$$

then, for the next interval we select the feature function with the lowest cumulative loss until now, i.e., $f = \arg\min_{f \in F} W_l^f(t)$.

Randomized Method: Instead of picking a feature function using cumulative loss as in the previous approach, in this method we select a feature function using a probabilistic approach. Let $\mathcal{P}_l^f(t)$, such that $\sum_{f \in F} \mathcal{P}_l^f(t) = 1$, be the probability of choosing a feature function from F for location l at time t. We initialize these probabilities to $\frac{1}{|F|}$. The weight $W_l^f(t)$ for every location-feature function is then used to determine probabilities for the next iteration. Like before, this weight is updated during every iteration as:

$$W_l^f(t) = W_l^f(t-1) \,\boldsymbol{.}\, \beta^{(1-\mathcal{A}_l^f)}$$

where, $\beta \in [0,1]$. By using this function of β, as the accuracy for a feature function decreases the weight corresponding to that function decreases. The probability for choosing a feature function is then updated as:

$$\mathcal{P}_l^f(t+1) = \frac{W_l^f(t)}{\sum_{f \in F} W_l^f(t)}$$

6.7 Experiments

We now evaluate the feature functions along with the learning algorithms described in this section. In the first set of experiments we analyze performance of

Greedy	**Local** **(Spatial)**	**Hashtag Trans.** **(Community)**	**Actual Hashtags** (% of hashtag occ.)
cgi2011	teamenzomusic	**cgi2011**	faze3 (0.29)
	takewallstreet	**dudesthatsayno*****	terriblenamesfor*** (0.29)
	cgi2011	**terriblenamesfor*****	cgi2011(0.14)
	miscellaney	foino20desetembro	dudesthatsayno*** (0.14)
	epatcon	**takewallstreet**	takewallstreet (0.14)

Accuracy = 0.20
Impact = 0.10
	Accuracy = 0.40
	Impact = 0.29
		Accuracy = 0.80
		Impact = 0.71	

Table 6.3: Top hashtags identified using different feature functions for New York on September 20, 2011 at 20:00.

feature functions using accuracy and impact, and then the analyze the effect of varying various learning parameters. We then evaluate some characteristics of the learning algorithms. For the experiments we use the dataset described in Section 6.3.

6.7.1 Performance of Learning Algorithms

In this section, we evaluate the performance of the feature functions and the learning algorithms using the metrics – accuracy and impact – described earlier in the section. We start by evaluating the performance of the the feature functions and the learning algorithms on fixed parameters and then evaluate the performance of the learning algorithms by varying parameters like number of top hashtags (k), the length of learning window and the length of prediction window.

An example of how the methods are evaluated, using evaluation metrics, is shown in Table 6.3. In this example, we predicted the subset of hashtags for New York on September 20, 2011. We predicted these hashtags at 20:00 UTC for the next 2 hours using a learning window of 6 hours. The first 3 columns show the prediction made by the 3 feature functions – greedy, local spatial affinity and community affinity

Prediction Method	**Accuracy**	**Impact**
Greedy	0.55	0.55
Global	0.64	0.64
Local	0.60	0.60
Common Hashtags	0.62	0.62
Hashtag Trans.	0.63	0.63
Linear Regression	0.32	0.32
Deterministic Method	**0.68**	**0.69**
Randomized Method	0.68	0.67

Table 6.4: Performance of various feature functions and learning algorithms

based on hashtag transmission. The last column shows the best set of hashtags or the gold set. The hashtags in bold indicate that they were one of the correct hashtags predicted. In this example, we observe one of the drawbacks of greedy approach – its inability to predict hashtags which it hasn't observed yet locally (in NY). The feature function using local spatial affinity does slightly better, in the sense it predicts mostly local hashtags, but misses out on hashtags that are popular globally like dudesthatsayno***, terriblenamesfor*** and so on. On the other hand, the feature function using community affinity based on hashtag transmission predicts 4 of the 5 hashtags correctly and performs the best. We also see that the performance of the feature function measured using the evaluation metrics we defined gives an indication of their actual performance.

We then evaluated the performance of all the feature functions and learning algorithms as shown in the example. We evaluated the methods using $w_p = 2$ hours, $w_l = 6$ hours and $k = 10$. The performance of the methods is shown in Table 6.4. Among the feature functions, we observe that the function that uses global spatial affinities performs the best. It has an accuracy and impact of 64%, implying that this method on average predicts 64% of hashtag occurrences for 2 hours in future correctly. In addition, as expected, the learning algorithms perform better than the individual feature functions with the method that uses reinforcement learning performing the best. The performance of this method could be attributed to the fact that it learns the best feature function for a location and uses that during predictions.

Performance With Varying k: For this experiment, we evaluated the performance on various learning algorithms by varying the number of top hashtags (k). We set the learning window length $w_l = 6$ hours, prediction window length $w_p = 2$ hour and then varied the value for k. The results of this experiment evaluated using accuracy and impact are shown in Figure 6.5(a) and Figure 6.6(a) respectively. The figures show the performance of the ranking algorithms as we vary the value of k from 1 to 25.

As described before, accuracy measures the similarity between the set of hashtags selected by our algorithms and the best set of hashtags for that interval, while impact measures how close we are to the best possible algorithm when it comes to the number of observed hashtag occurrences. We know that the distribution of hashtags at a location follows a zipfian distribution with few trails accounting for most occurrences. Hence, the problem of selecting top$-k$ hashtags becomes harder when k is small. The result of this distribution is reflected in the performance of our ranking algorithms as well, where we observe that the performance of you algorithm improves as the value of k increases. The zipfian distribution also explains the flattening of the curve after around $k = 10$. The hashtags selected by the algorithms after this value of k don't result in significant increase in impact as the observed occurrences of these hashtags is small, resulting in the flattening of the curve. Of the the learning algorithms, the algorithms that used reinforcement learning performed better than the algorithm that used linear regression to estimate the value of expected hashtag occurrences.

Performance With Varying w_l: To evaluate the performance of our ranking algorithms for varying lengths of learning time window, we set the prediction time window $w_p = 2$ hours and $k = 10$. We varied w_l from 1 hour to 10 hours in 1 hour intervals. The results from this experiment using accuracy is shown in Figure 6.5(b) and using impact is shown in Figure 6.6(b).

We observe that the performance of the learning algorithms that use reinforcement is better than the algorithm that uses linear regression. But, there is no significant difference between the two methods that use reinforcement learning. Initially, as the length of learning window increases we see in improvement in accuracy (and impact) for all the algorithms. But accuracy beings to level out as the length of estimation window continues to increase. We believe the performance of the algorithms improves during initial increase in learning window because with a longer window they are able to analyze larger number of hashtag occurrences which helps them make better decisions during prediction. But, as the window continues to increase they observe older hashtags propagations, which results in evening out or even decreasing performance. The window that is best suitable for estimation might depend on the network on which the social network are propagating and the nature of hashtag themselves. In case of hashtag propagation on Twitter we found a window of 6 hours was best suited for hashtag prediction.

Performance With Varying w_p: We next evaluated the performance of our learning algorithms for varying lengths of prediction time window. We set the learning time window $w_l = 6$ hours and $k = 10$. We then varied w_p from 1 hour to 10 hours in 1 hour intervals. The results from this experiment using accuracy is shown in Figure 6.5(c) and using impact is shown in Figure 6.6(c).

Like in earlier experiments we observe that learning algorithms that use reinforcement perform better than the linear regression algorithm. In particular, we observe that the performance of the algorithms peaks when the prediction window is 2 hours and then decreases with the increase in length of prediction window. This result shows the sensitivity of the prediction window, because unlike w_l which had a region in which the performance didn't change, in case of w_p the performance of the model decreases almost linearly with time.

6.7.2 Learning Analysis

In this section we analyze learning algorithms in detail. We first analyze the impact scores obtained using these algorithms and then analyze the rate at which the learning algorithms assign feature function to locations. We then analyze these algorithms further by defining a metric called flipping ratio which measures the uncertainty of a learning algorithm in assigning feature functions.

Analysis of Impact Scores: We next analyze the impact scores for all the locations in our dataset. For this analysis we use impact scores obtained using the three algorithms that were compared in the previous section. Every location is assigned the best algorithm specific to it. We divided all the locations into 4 regions – United States (0.33%), Europe (0.34%), South America (0.25%) and South-East Asia (0.08%). The number in bracket indicates the percentage of locations in the region. The distribution of the algorithms is shown in Figure 6.7. In spite of varying number of locations in each region we observe that distribution of learning models is similar. All the regions have almost equal number of locations that prefer either deterministic or randomized algorithm and a small number of locations prefer linear regression.

The distribution of impact scores and its complementary cumulative distribution function is shown in Figure 6.8(a) and Figure 6.8(b) respectively. As described earlier impact in a way measures how close the learning algorithm selected for a location is close to the ideal algorithm that can be designed for that location. So, a impact of 1.0 signifies the algorithm as good as the best algorithm. We observed that more than half of the locations, for which we made predictions, we were able to achieve an impact of at least 0.70.

Analysis With Learning Rate: In this experiment we compare the rate at which the two reinforcement algorithms, we described in Section 6.6.2, learn feature function to be assigned to a location. The result of this experiment is shown in Figure 6.9. In this figure, the learning time is shown in x-axis and the percentage of locations that flipped their decision in the current interval is shown in y-axis.

We observe that the deterministic algorithm is faster than the randomized algorithm. The flipping nature of these algorithms could be attributed to the way in which they select feature functions. The randomized algorithm selects a feature function based upon probabilities estimated from the feature function weights while the deterministic algorithm is much simpler in the sense it makes a decision based upon the cumulative loss. These probabilities are non-zero for more than one feature function resulting in the algorithm flipping more. This issue is not observed in case of the deterministic algorithms making it much more stable and hence faster. In spite of the simple nature of deterministic algorithm we observe that its performance as better than that of the randomized algorithm. For hashtag propagation in Twitter we saw that we were able to assign feature function to locations using about a weeks data (flatting of red curve in Figure 6.9).

Analysis With Flipping ratio: We first describe flipping ratio and then analyze the learning algorithms using it. In our experiments test set is broken into time intervals of equal size. The learning algorithms select a feature function every interval. Then, flipping ratio measures the uncertainty of a learning algorithm by determining the number of times the algorithm changes its decision from that made in previous interval. It is defined as:

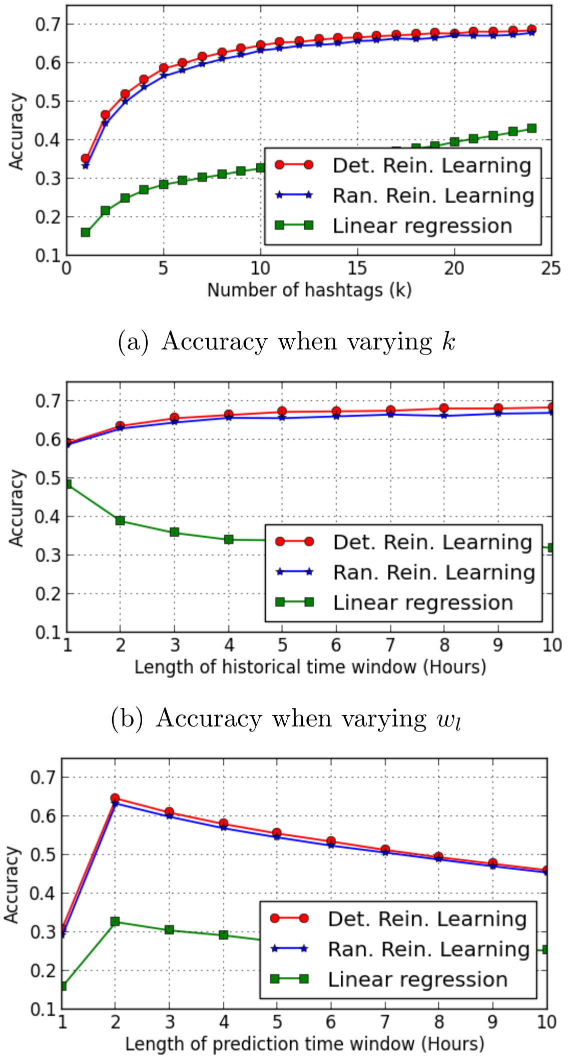
$$\text{Flipping Ratio} = \frac{\text{\# of decision changes}}{\text{\# of intervals in test set}}$$

where, an ideal learning algorithm with flipping ratio 0.0 will pick a feature function for a location in its first attempt, while the worst learning model with flipping ratio 1.0 will change its decision every interval.

We analyzed the correlation between the density of location and its flipping ratio. Since, we can't get the exact density for every location, we assume hashtag occurrences at a location as an indicator of the actual density. One of the issue with this assumption is that hashtag occurrence counts might not be a good indicator of actual density. For example, there could be dense locations with poor Internet connectivity resulting in low occurrences, while college towns with low density might have large number of occurrences. But, this assumption doesn't impact applications using hashtag subset selection, because the hashtags selected by our models are still reflective of the user activity online and not the actual density. The correlation between density of a location and its flipping ratio is shown in Figure 6.10. We see that flipping ratio decreases with increase in density of a place. In other words, the ability of a learning algorithm to assign feature function to a location increases as the number of hashtag occurrences at that location increases. This is an important result because, the earlier and more accurately we can assign feature function to a location with high density the better performance of our algorithm is.

6.8 Conclusion

In this section, we proposed and evaluated approaches that predict where and when a online meme will be popular. In particular, we developed models based on the two competing hypotheses of information spread over geo-spatial networks – spatial affinity and community affinity. We then evaluated these models over a collection of 755 million geo-tagged Tweets and found a model that can predict future hashtags occurrences with a 70% accuracy. In our future work, we are interested in analyzing how these approaches scale under large amount of data arriving at rapid rate.

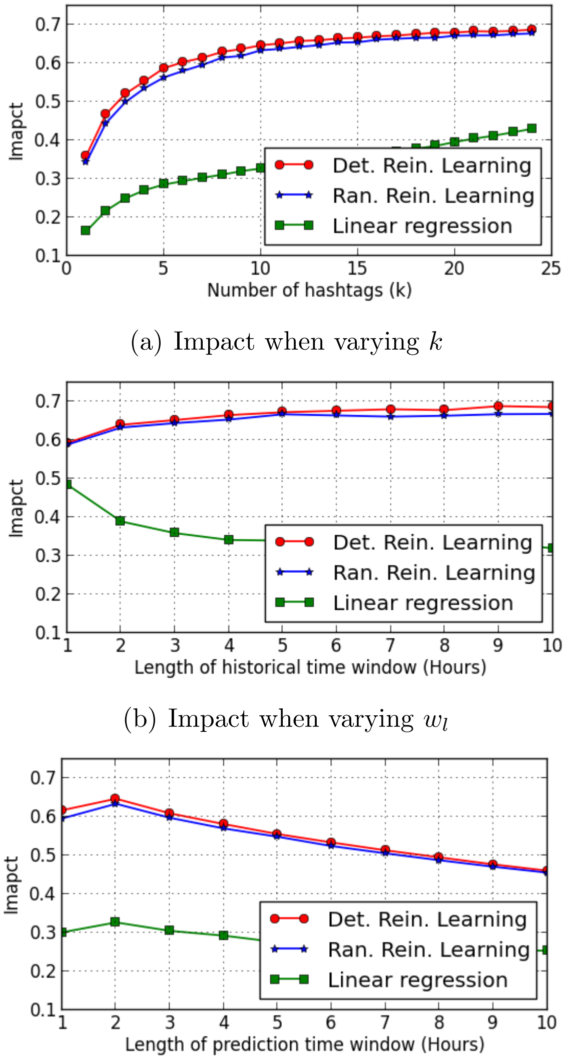


(a) Accuracy when varying k

(b) Accuracy when varying w_l

(c) Accuracy when varying w_p

Figure 6.5: Ranking Model Performance (Accuracy)



(a) Impact when varying k

(b) Impact when varying w_l

(c) Impact when varying w_p

Figure 6.6: Ranking Model Performance (Impact)

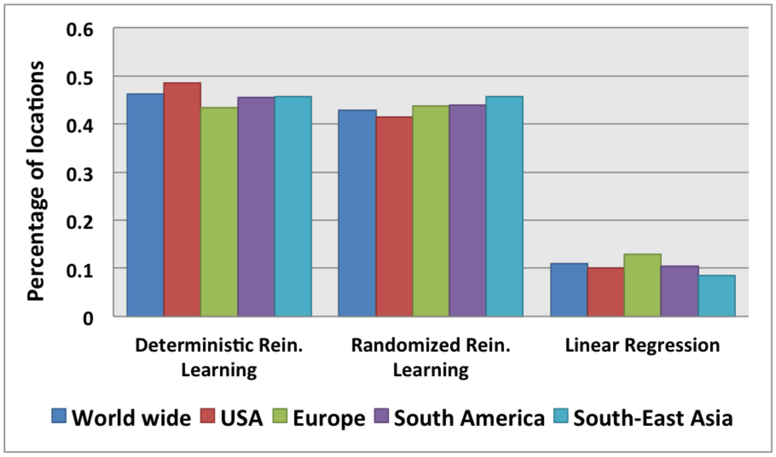
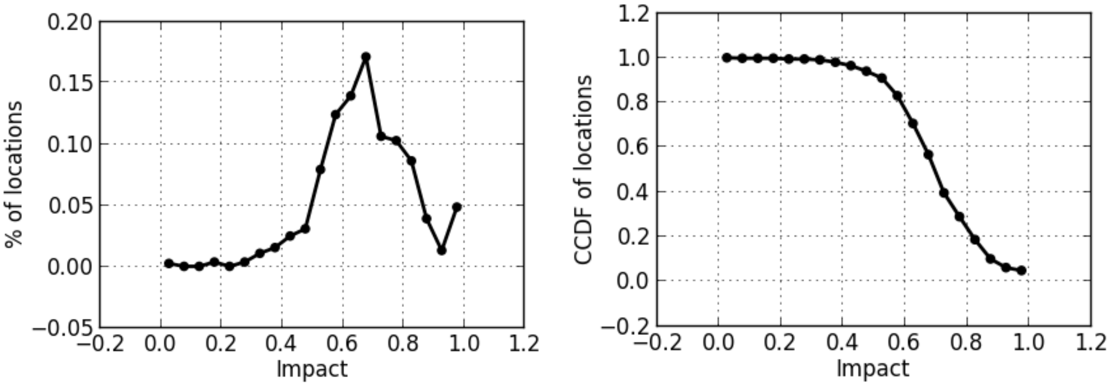


Figure 6.7: Distribution of preferred learning algorithm in various locations by geographical areas (Impact).



(a) Distribution of impact scores

(b) CCDF of impact scores

Figure 6.8: Analysis of impact scores for various locations. Using our learning algorithms we were able to achieve a impact of at least 60% for more than 80% of the locations.

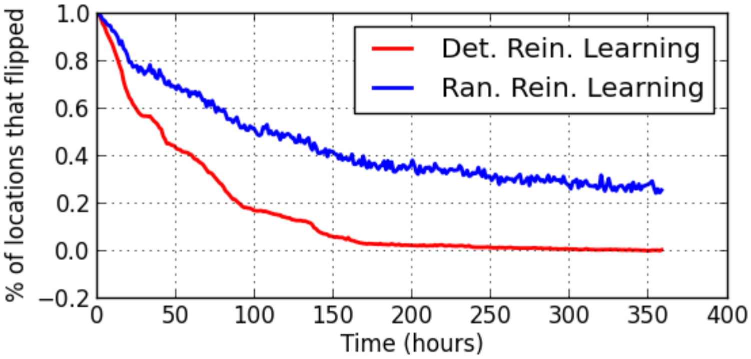


Figure 6.9: Learning rate comparison

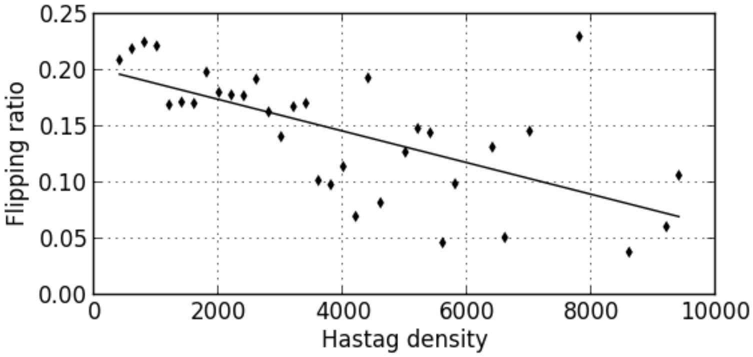


Figure 6.10: Flipping ratio Vs location density

7. REAL-TIME CLASSIFICATION OF SOCIAL TRAILS*

In this section, we develop methods to classify social trails. In particular, we study the problem of expert-driven topical classification of social trails in time-evolving streams like Facebook status updates, Twitter messages, and SMS communication. While high-level topics in these streams may be fixed (e.g., sports, news), the content associated with these topics is typically less static, reflecting temporal change in interest as these streams evolve (e.g., tweets about the Olympics wane, while tweets about the World Cup rise in popularity). Coupled with this rapid concept drift, short messages themselves provide little contextual information and result in sparse features for effective classification. With these challenges in mind, we present an expert-driven framework for time-aware topical classification framework of short messages. Three of the salient features of the framework are (i) a novel expert-centric classifier; (ii) a sliding-window training for adaptive topical classification; and (iii) a suite of enrichment-based methods (lexical, link, collocation) for overcoming feature sparsity in short messages.

7.1 Introduction

One of the key challenges for making sense of these high-volume short message streams is in organizing these *unstructured* social streams into *structured* categories of interest. For example, several recent efforts have begun the study of Twitter message classification in the context of information filtering [70], news aggregation [66] and business specific mining [79]. In many of these cases, however, mapping from unstructured social streams to structured categories of interest may lead to

*Part of this section is reprinted with permission from "Expert-Driven Topical Classication of Short Message Streams" by Krishna Y. Kamath and James Caverlee, 2011. Conference on Social Computing. 3rd. Copyright 2013 by Institute of Electrical and Electronics Engineers (IEEE).

errors and poor quality identification of relevant messages due to a number of key challenges:

- The rapid evolution of social streams, so that important keywords associated with a concept one day may not correspond to the same concepts the next day. To illustrate, Figure 7.1 shows how the prevalence of the keyword "healthcare" varied on Twitter across several categories during the healthcare debate (details described later in the section). Note that during the month of March (weeks 9 to 12) the Senate was debating the healthcare bill leading to many mentions of "healthcare" in politics; at other times, "healthcare" was associated with business-related messages and of course, health-related messages.

- The inherent error-laden and lack of context in many messaging systems that restrict the number of characters (140 characters, in the case of Twitter). As an example, consider the message – "Almst over da Flu..stayin in all weeknd" – which contains shortened words and misspellings.

- A mismatch between the language in use by participants and the language expected by the classification framewrok (e.g., the use of emergent hashtags, colloquialisms) as in an example tweet describing an earthquake "Ahh!! :S tremble. Walls cracking!! #timetoleave".

Together, this coupling of rapid concept drift, lack of contextual information, and sparse feature representation present strong challenges to effective and ongoing topical classification of short message streams. With these challenges in mind, we present an expert-driven framework for time-aware topical classification framework of short messages. The key insight driving the framework is the reliance on category-specific *experts*, whose streams themselves may serve as prototypes for learning generalized categorical models for robust stream classification. We show how these

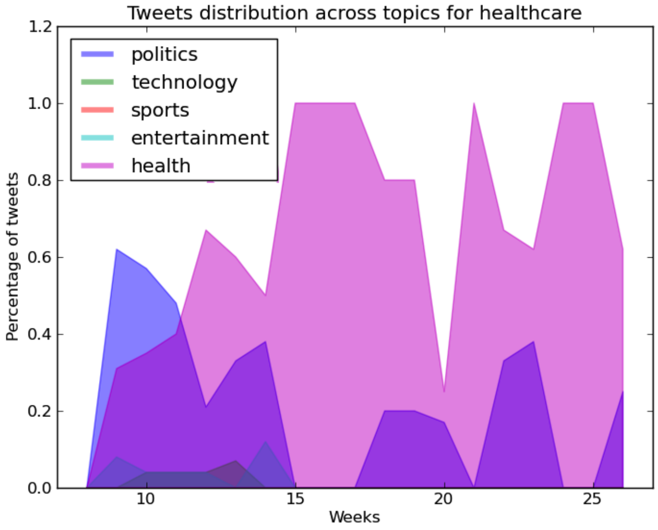


Figure 7.1: Prevalence of the term 'healthcare' across domains from March 2010 to July 2010.

expert streams may seed classification, and we propose a sliding-window training approach for adaptive topical classification. Additionally, we explore techniques for augmenting short messages using feature-based, link-based and collocation expansion. Through experimental study over Twitter, we find good performance of the proposed method for ongoing expert-driven topical classification of short message streams.

7.2 Problem Statement and Setup

In this section we present the overall framework of our study of expert-driven topical classification over short message streams. We begin with a discussion of the problem, and then introduce the data and baseline classifier used in the rest of the section.

7.2.1 Problem Statement

While a domain model may identify an arbitrarily complex concept hierarchy, we focus in this section on a simple one-level hierarchy corresponding to general high-level topical categories. We selected four high-level categories for this study that are generally well-represented in current popular social messaging systems: *politics*, *technology*, *sports* and *entertainment*. For each category, the system takes as input a set of *expert* accounts and their messages. These experts are intended to be representatives of the category, although not all of their messages may actually belong to a single category. For example, a sports-themed account may intersperse entertainment and politics messages in their stream of mostly sports-related messages. In practice we will only be able to identify a small number of expert accounts relative to the large body of actual accounts in a system. Given a set of categories and a list of expert accounts, we seek to identify messages over time that map to these categories. We refer to this as the problem of *expert-driven topical classification of short messages in time-evolving streams.*

7.2.2 Data

For this study, we require a collection of time-stamped short messages from across a number of different categories. While there are large benchmark collections of Web pages, email messages, and other longer-form documents, we are unaware of any existing topically-segmented short message collections. Hence, we collect a "ground-truth" domain-specific Twitter stream by identifying prominent accounts for the 4 domains – technology, entertainment, politics and sports – using a snowball sampling approach described in [77]. The output of this snowball sampling method is for each category an ordered list of accounts, ordered by their significance within that category (the details are omitted here, but explained more fully in [77]). The seed accounts

Domain	Total Messages	Messages per day
Politics	30,658	143
Technology	21,880	102
Sports	67,782	316
Entertainment	38,477	179

Table 7.1: Data distribution per domain

selected for snowball sampling for each category is shown in Table 7.8 at the end of the section. Using these seed accounts for each domain we select the top 1,250 accounts and use the “follow” parameter of the filter method from Streaming API to generate a domain specific stream of tweets. Using this approach, we collected a total of 209,046 messages between March and April 2011. The breakdown per domain is shown in Table 7.1.

7.2.3 Topical Classification with MaxEnt

Given a message from a social messaging system, we aim to automatically determine its appropriate category through an analysis of the text in the message itself. While many text classifiers are possible (e.g., Naive Bayes, Support Vector Machines), we focus in this section on maximum entropy (MaxEnt) classification [55], which has been shown to efficiently model domains in which information is sparse (as in the case of short messages). MaxEnt is based on the maximum entropy principle [14] and has been widely used for text classification [60]. We will now describe the maximum entropy principle in terms of text classification.

Consider a document (short message) d that belongs to class y in a training set of labeled documents. Generally, in text classification, terms in the documents are represented as features. So, let x be a term in d. Then we can define a feature function $f(x, y)$ as an indicator random variable.

$$f(x,y) = \begin{cases} 1 & If\ x\ is\ in\ document\ of\ class\ y \\ 0 & Otherwise \end{cases}$$

From the training set we can calculate the empirical probability distribution $\tilde{p}(x,y)$ of observing x in documents of class y. Using this we can determine the empirical expected value of f.

$$\tilde{p}(f) = \sum_{x,y} \tilde{p}(x,y) f(x,y)$$

When the ideal classification model $p(y|x)$ is known, we can use the empirical distribution of x, $\tilde{p}(x)$ (calculated from the training set), to determine the expected value of f as:

$$p(f) = \sum_{x,y} \tilde{p}(x) p(y|x) f(x,y)$$

Now, given a set of feature functions $F = \{f_1, f_2, \ldots, f_n\}$, one for every term, and the space of all probability distributions P we can define $C \subset P$, as the set of distributions which give the same expected value of f as the empirical value of f obtained from the training set.

$$C \equiv \{p \in P|\ p(f_i) = \tilde{p}(f_i)\ \ for\ i \in \{1, 2, \ldots n\}\}$$

Of all the models (distributions) in C, we have to pick the model that gives the most uniform distribution. Hence, we can use conditional entropy $H(p)$ to optimize the solution.

$$H(p) \equiv -\sum_{x,y} \tilde{p}(x) p(y|x) log p(y|x)$$

The maximum entropy principle states to pick the the distribution $p_\star \in C$ that yields the maximum entropy $H(x)$:

$$p_\star = \arg\max_{p \in C} H(x)$$

For text classification, $p_\star$ gives us the model from which the probability that document d, which contains a term x, belongs to a class y can be calculated using $p(y|x)$. In this way, we can assign short messages to one of the four categories.

7.3 Overall Expert-Driven Approach

Toward bridging the gap between unstructured social messaging streams and structured categories of interest, we must first identify a set of candidate expert accounts associated with each category – these expert accounts serve as prototypes of what we expect to see from a particular category. While the particular expert-selection criteria may vary across domains and application setting, we adopt a baseline approach where we select as *experts* the top 125 accounts in each domain as ordered by the snowball sampling approach described in Section 7.2.2.

7.3.1 Sliding Window Training

Given an appropriate selection of expert streams, to abate the effects of rapid concept drift we propose to train a classifier over a *sliding window* to capture the day-to-day and hour-to-hour changes in the concepts associated with a particular category. Using a fixed period of days, we could monitor all messages posted by the expert accounts, build a classification model based on these messages, and then classify all new messages based on this model. For example, if today is the 11^{th} day of March, then we could build a classifier over the prior eight days of messages posted by the pre-seeded expert accounts (from 3^{rd} March to 10^{th} March) and apply this new model to all messages encountered. Moving to the following day, the classification models could be updated with the sliding window (now from 4^{th} March to 11^{th} March), and so on and so on. In this way, the classification decisions are based primarily on concepts that are recently reflected in the social messaging system, rather than being tied to immutable keywords.

Of course, there are a number of open questions: (i) What is the best size of a sliding window? Choosing a very small window may perform well on bursty events within a category (e.g., a particular football game within the domain "sports"), but more poorly on longer-lived themes. (ii) Is there enough feature density (i.e., keywords) in each expert stream to produce robust topical classifiers? (iii) How can this feature sparsity be overcome in a lightweight manner?

7.4 Short Message Enrichment

Even with a dynamic sliding window classifier in place, we still face one of the key challenges to content-based classification of short messages – the problem of limited features found in these messages. Whereas traditional web page and document classification tasks have typically focused on feature selection for reducing the many available word-based features to identify a smaller set of highly-valuable distinguishing features, in short message classification we take an alternate approach to enrich the sparse messages with additional features. Concretely, we explore three general approaches for overcoming feature sparsity in short messages: (i) lexical-based, in which features in short messages are increased by applying lexical feature expansion techniques based on the content within the message; (ii) external-based, in which externally-derived features like part-of-speech and URL features extracted from links embedded in short messages are used to augment the feature representation; and (iii) collocation-based, in which the terms in a message are associated with related terms (collocations) from other messages, and these related terms are added as features to the original message.

7.4.1 Lexical-Based Enrichment

To overcome the sparsity of feature set in short message classification we can use lexical feature expansion techniques. We use bigrams, trigrams and orthogonal sparse bigrams to increase features. The details of these techniques are given below:

- *Character n-grams*: Using this technique n consecutive characters in the message are used as features. For example, in the case of character 3-grams for the message "Go yankees!", we use "go ", "o y", " ya", etc., as features. The intuition is that these n-grams may overcome problems in spelling, in shortened text, and other artifacts of short messages.

- *Word n-grams*: Similar to character n-grams, in this technique n consecutive words in the original message are used as features. For example in the case of bigrams, for the message, "Mark Teixeira removed from New York Yankees roster", we use "mark teixeira", "teixeira removed", "removed from", "from new", "new york", "york yankees" and "yankees roster" as the features. Similarly, we can obtain features for trigrams as well by considering every three consecutive words as features.

- *Orthogonal Sparse Word Bigrams:* Following Cormack [16], this technique generates as features every pair of words that are separated by 3 or fewer words. For example, for the message used in word n-grams we use "mark (0)teixeira", "mark (1)removed", "mark(2) from", "teixeira(0)removed", "teixeira(1)from", "teixeira(2)roster", "removed(0) from", "removed (1)roster" and "from(0) roster" as features.

7.4.2 External-Based Enrichment

In this approach we overcome feature sparsity in short messages by augmenting each message with features extracted from an external resource. Specifically, we consider two approaches: link-based and part-of-speech-based.

- *Link-based*: Short messages often contain URLs in them and in many cases an individual URL linking to a webpage contains information that describes the page. We call this information collected from the raw URLs the link meta information.

  For example, consider the URL:

  http://www.nytimes.com/2010/07/27/sports/football/
  27concussion.html?ref=sports

  By just reading the URL we can understand that the page is a sports page about football that talks about concussions. We can extract the terms sports and football from the URL and enrich the short message with it. For URLs that are shortened using service like bit.ly, goo.gl etc., we expand the actual link pointed by the shortened URL and extract meta information from the long-form URL.

- *Part-of-speech*: In a given short message, identifying nouns can give us a good understanding of the message topic. So, in our analysis we tagged terms in a message with their corresponding part-of-speech (POS). We used the POS tagging feature in NLTK Python toolkit [50] and filtered words which were not tagged as nouns.

7.4.3 Collocation-Based Enrichment

The third expansion technique considers words that are associated with the words in a message. We identify associated words by examining collocations from across other expert accounts within a category and from what we refer to as “affiliate” accounts (described more fully in the experiments section). A collocation is “an expression consisting of two or more words that correspond to some conventional way of saying things” [52]. Examples of collocations are *kobe bryant, boston celtics*, etc. Intuitively, a short message may refer to some aspect of a concept (e.g., “kobe”), but due to the space limitation may not include other related terms (e.g., “bryant”, “lakers”). By identifying collocations, we can enrich a single message with additional terms, but perhaps at the cost of introducing noise terms.

Concretely, we limit ourselves in this section to collocations consisting of two words only. To identify collocations we first need an association measure between words. Association measures, are mathematical formulae, used to measure the closeness between the words of a phrase. This measure is used to rank the pair of words. The measure is based on the count of occurrences of words and co-occurrences between pairs of words. There are various association measures starting from plain frequency of occurrence, to measures based on information theory like mutual information and heuristic methods.

In [52], the authors have illustrated the problems with association measures that use frequency or variance to determine collocations. They also show mutual information is not very suitable to identify collocations. Hence, to determine collocations in this section we will be using two asymptotic hypothesis test methods: *Pearson’s chi-squared test (χ^2)* and *Dunning’s log-likelihood ratio test.* Generally it is observed that the log-likelihood test is more useful in determining collocations on sparse data

Observed Frequencies	$V = v$	$V \neq v$
$U = u$	O_{11}	O_{12}
$U \neq u$	O_{21}	O_{22}

Table 7.2: Observed frequencies.

Expected Frequencies	$V = v$	$V \neq v$
$U = u$	$E_{11} = \frac{R_1 C_1}{N}$	$E_{12} = \frac{R_1 C_2}{N}$
$U \neq u$	$E_{21} = \frac{R_2 C_1}{N}$	$E_{22} = \frac{R_2 C_2}{N}$

Table 7.3: Expected frequencies.

compared to the χ^2 test.

In hypothesis testing we formulate a *null hypothesis* H_0 that two words, u and v, are independent of each other. Let p be the probability that the event H_0 occurs. We then calculate p from the frequencies of u and v in the data-set. If p is lower than a *probability threshold* α, say $p < 0.05$, we reject the hypothesis that u and v are independent and accept the word pair as collocation.

We use two types of frequencies to calculate the association measures. As shown in Table 7.2, for word pair u and v in the data-set we define observed frequencies O_{11} as the number of times u, v appear together, O_{12} as the number of times u occurs without v, O_{21} as the number of times v appears without u and O_{22} as the number of time u and v don't occur at all. To calculate the expected frequencies, from observed frequencies we calculate the occurrence of u, $R_i = \sum_{j \in (1,2)} O_{ij}$, occurrence of v $C_i = \sum_{j \in (1,2)} O_{ji}$ and total number of words $N = \sum_{i \in (1,2)} R_i + C_i$. With these values E_{ij} for $i, j \in (1, 2)$ is calculated as shown in Table 7.3.

Contingency Table	$V =$ **lakers**	$V \neq$ **lakers**
$U =$ **kobe**	150	932
$U \neq$ **kobe**	12,593	14,307,668

Table 7.4: Contingency table for kobe and lakers.

7.4.3.1 Pearson's chi-squared test (χ^2)

This test compares the observed (O_{ij}) and expected (E_{ij}) frequencies. If the difference between them is large H_0 is rejected. The method is similar to calculation of mean-square error.

$$\chi^2 = \sum_{i,j} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

The value calculated from the above equation has χ^2 distribution. In the case of a 2x2 table, we have one degree of freedom, hence at $\alpha = 0.05$ the critical value is $\chi^2 = 3.84$. For a pair of words u and v, we calculate the χ^2_{uv} value. If $\chi^2_{uv} > 3.84$ we can reject H_0 for the pair of words and accept them as collocations. For example, in Table 7.4, we show an example of contingency table for words *kobe* and *lakers*. The χ^2 value for this pair is approximately 27,155, which is greater than 3.84. Hence, the words can be accepted as collocations.

7.4.3.2 Dunning's log-likelihood ratio test

In practice, O_{11} is very small compared to N, due to the inherent sparseness of social messaging streams. In such cases, with a highly skewed contingency table, Dunning [23] has shown that the log-likelihood measure can better than the χ^2 test. In this case, the log-likehood is:

$$Log - likelihood = -2\log\frac{L(O_{11},\ C_1,\ r).L(O_{12},\ C_2,\ r)}{L(O_{11},\ C_1,\ r_1).L(O_{12},\ C_2,\ r_2)} \tag{7.1}$$

where,

$$L(k,\ n,\ r) = r^k(1-r)^{(n-k)}$$
$$r = \frac{R_1}{N},\quad r = \frac{O_{11}}{C_1},\quad r = \frac{O_{12}}{C_2}$$

As in the previous association measure, the log-likelihood measure ratio has an asymptotic χ^2 distribution. For the example in Table 7.4, the log-likelihood ratio for *kobe* and *lakers* is 1291. By using these two different techniques, we can observe the impact of collocation-based augmenting of short messages on classification performance.

7.5 Experimental Study

In this section, we present a comparative study of the time-aware topical classification framework for short messages. We use the dataset of 209,046 messages across four categories, collected during March-April, 2011. Using the top-125 accounts per domain as the seed experts, we test the developed topical classifiers over a test set consisting of the *bottom* 125 accounts per domain (out of 1,250), meaning that these test accounts are only loosely-related to the categories of interest and non-overlapping with the expert accounts.

7.5.1 Metrics

To evaluate the quality of a topical classifier over short message streams, we consider a variation of the area under the Receiver Operating Characteristic (ROC) curve called the *M-value*.

M-value: The area under Receiver Operating Characteristic (ROC) curve is a widely-used metric to measure the performance of classifiers. But, it is appropriate only in binary classification problems and hence cannot be directly applied to multi-class classification problems. So, in this section, since we are dealing with a multi-class classification problem, we use a metric which is an generalization of the ROC metric used in binary classification. We use the popular *M-value* metric proposed by Hand and Till [36], that extends the area under the curve definition to the case of more than two classes by averaging pairwise comparisons.

Given a set of classes $C = \{c_1, c_2 \dots c_k |\ k > 2\}$ and a document d in the test set, the classification algorithm gives us an estimate of the probability of the document belonging to any class c, $P(c|d)\ \forall c \in C$. Given this we can calculate $\hat{A}(i|j)\ \forall i, j \in C$. $\hat{A}(i|j)$ is defined as the probability that a randomly drawn member of class j will have a lower estimated probability of belonging to class i than a randomly drawn member of class i. Similarly, we can calculate the value of $\hat{A}(j|i)$ as well. Note that in case of binary classifiers $\hat{A}(0|1) = \hat{A}(1|0)$, while this is not true in the case of multi-class classifiers, i.e. $\hat{A}(0|1) \neq \hat{A}(1|0)$. We then calculate the separability between any two classes as $\hat{A}(i, j) = [\hat{A}(i|j) + \hat{A}(j|i)]/2$.

The overall separability for all the classes – the M-value – is given by the average of all the values of $\hat{A}(i, j)$:

$$M = \frac{1}{\binom{|C|}{2}} \sum_{i<j} \hat{A}(i, j)$$

A higher M-value indicates a “better” classifier.

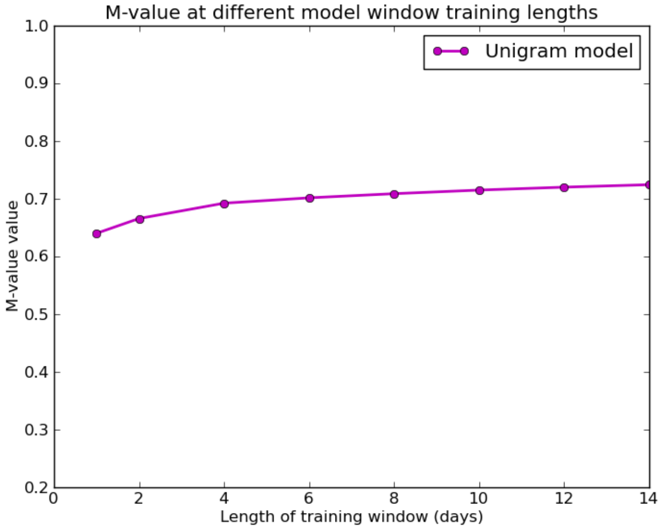


Figure 7.2: M-value at different training-window sizes.

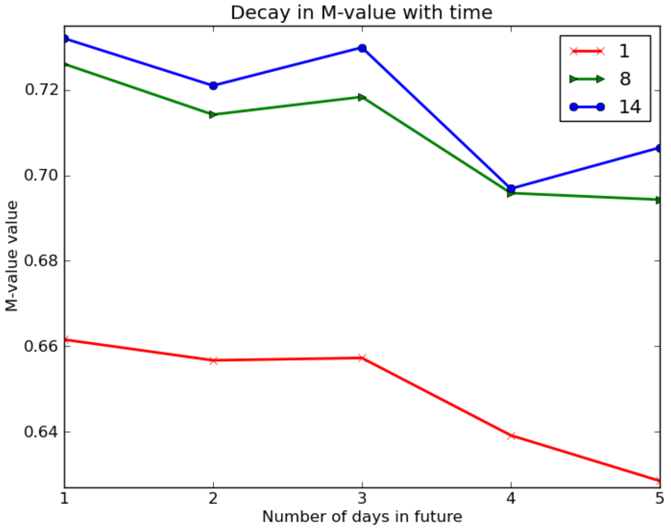


Figure 7.3: Diminishing M-value of the classifiers with time at different model lengths.

7.5.2 Sliding Window Length

We begin the experimental study by examining how the size of the training window impacts the quality of categorization. We first try different window lengths and observe the length at which the M-value is maximum. We then use the models trained on different window lengths to see how they perform over time.

Different Window Lengths: The sliding window approach we advocate requires that we identify a set of expert accounts to serve as our prototypes for each category. For fairness, we train on the messages in the gold set for days leading up to but not including the test day of messages.

We trained a MaxEnt classifiers, using unigram features, on different training-windows to determine the optimum length. In Figure 7.2, we show the performance of the classifiers that were trained on a window length from 1 to 14 days. We observe that the M-value is lowest with only a single day of training; this indicates that the concepts introduced on a single day are not representative of the overall theme. The curve flattens around the 8^{th} day, indicating that about a week's worth of messages are necessary to capture the main concepts. We also notice that classifier that is trained for around 8 days yields almost the same accuracy as a classifier trained for 14 days, indicating that longer window sizes do not necessarily lead to large gain in classification accuracy.

Classifier Decay: We next investigate how long after a classifier has been built it is still effective. We know that as new messages are observed and newer concepts introduced the accuracy of an older classifier decreases. We refer to this decrease in M-value of a classifier with time as classifier decay. A good classifier should decay relatively slowly, meaning that the essential characteristics of a category have been learned. To analyze classifier decay, we took classifiers that were trained on 1, 8 and 14 day windows, and used them to classify tweets. This is shown in Figure 7.3. We

Description	M-value
Unigrams	0.71
Character bigrams	0.67
Bigrams	0.49
Orthogonal sparse bigrams	0.54

Table 7.5: Lexical feature expansion

observe that the classifier trained on a 14 day window decays slowly.

The difference in decay can be attributed to the features that these different classifiers learn from the training set. A classifier trained for 14 days learns concepts that are spread over a longer duration of time while the 8 day classifier picks up concepts that occur for a shorter time. For example, a 14 day classifier may learn features related to MLB games, an event that happens over months, and not learn relatively short events like individual games during March madness, which happens on a single day. But the 8 day classifier is able to learn these events of shorter durations.

7.5.3 Short Message Enrichment

Based on the results in the previous section, we next evaluate the several approaches to short message enrichment where each classifier has been trained over an 8 day window. We begin by testing the performance of lexical feature expansion, as shown in Table 7.5. First, we can see that the unigram gives the best performance of all the lexical approaches. Hence, from now on for all the experiments we will be unigrams as features for classification.

To test the performance of the classifiers with collocation-based expansion, we append the messages in the training set with the collocations discovered using χ^2 and Dunning's log-likelihood. We use two different collections of messages to identify col-

Description	**M-value**
χ^2 (experts)	0.70
χ^2 **(affiliates)**	**0.74**
Dunning's (experts)	0.69
Dunning's (affiliates)	0.69

Table 7.6: Collocation-based expansion

Description	**M-value**
Unigrams	0.70
Nouns	0.66
Unigrams + link	0.71
Nouns + link	0.66

Table 7.7: Link and POS-based expansion

locations: (i) Messages from "experts": Top 125 accounts per domain (500 accounts); and (ii) Messages from "affiliates": Top 375 accounts per domain (1500 accounts). The affiliates are accounts outside of the top 125 accounts. The hypothesis is that by enriching messages with collocations from affiliate accounts, we may identify extra category-specific collocation terms that cannot be obtained from experts. For all four cases, the performance of the classifiers is shown in Table 7.6. Interestingly, we note that the performance is improved by using collocations obtained from affiliates.

We next test the two external enrichment approaches – part-of-speech tagging and link expansion. We see in Table 7.7 that the noun-based approach results in a smaller M-value than the unigram approach, indicating that the key distinguishing features for topical classification are most likely to be unigrams. Also, when introducing link-based information in spite of additional features we don't see any improvement in performance. This is quite encouraging, since extracting nouns and link information is expensive in a real-time application, and the result that unigrams can yield the

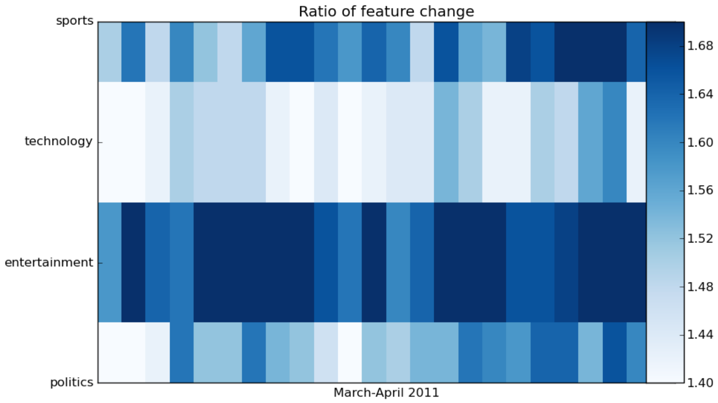


Figure 7.4: Ratio of change in top features every day.

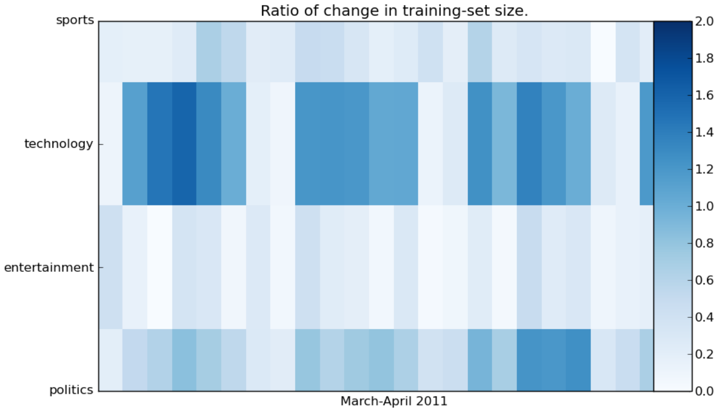


Figure 7.5: Ratio of change of training-set size.

best performance can motivate efficient classification algorithms.

7.5.4 Temporal Analysis of Classifiers

Finally, we are interested to explore the dynamics of expert-driven short message classification over time.

Ratio of feature change: We begin our investigation of the temporal dynamics by first determining the the ratio of feature change between intervals. For a given class $c \in C$, let $F_{c,t}$ be the set of features used by the classifier at time t. The ratio of feature change $R_{c,t} \in [0,1]$ for c in the time interval t is calculated as:

$$R_{c,t} = \frac{|F_{c,t_i} - F_{c,t_{i-1}}|}{|F_{c,t_{i-1}}|}$$

To calculate the ratio of feature change R_{c,t_i} we determine the top features for c in consecutive intervals, $F_{c,t_{i-1}}$ and F_{c,t_i}. We then determine the number of features that have been newly added in the interval t_i, $|F_{c,t_i} - F_{c,t_{i-1}}|$. Note, that the number of top features we get in each interval is a constant. Hence, the value of $R_{c,t}$ lies between 0 and 1. For a class that is very dynamic, with concepts (features) constantly changing, the value of $R_{c,t}$ is closer to 1, while in case of a static dataset the value is closer to 0.

To observe the ratio of feature change across the different classes, we used the classifiers generated during March-April, 2011. Each of these classifiers are trained on a window of 8 days of data and uses unigrams for feature representation.

We used a heat map to visualize the ratio of feature change, shown in Figure 7.4. The intensity of a block on the map is proportional to the value of change ratio. Using the heat map we can compare the dynamic nature of the different classes. We can make the following observations:

- All classes are colored almost throughout the month. This shows us the presence of rapid concept drift in short message streams.
- Sports is the most dynamic class with bright colored blocks almost throughout the month. This tells us that concepts in sports change on a daily basis.
- Technology and health are much more stable compared to sports.

Correlation between feature change and training set size: To continue with this analysis of ratio of feature change, we wanted to understand if there was a correlation between the training set size and feature change. A high increase/decrease in the number of training documents for a particular class between two intervals may have an impact on the features discovered in the interval. This correlation is not desirable because, it indicates the classifiers are over-fitting the training data and hence discovering inaccurate concept drifts.

Like ratio of feature change $R_{c,t}$, we calculate the ratio of change in training set size for each class between time intervals. We used the same data that was used in training the 31 classifiers which were used in the previous analysis. The heat map for the ratio of change in training set sizes is shown in Figure 7.5. As shown by the large number of light colored blocks, we don't observe large change in training set sizes between intervals. This indicates the training-set was uniformly distributed for the classes throughout the month. Also, a comparison of both the heat maps in Figure 7.4 and Figure 7.5, doesn't show any correlation between the ratios. The independence between the two ratios indicates that the features discovered by the classifiers correspond to actual concept drifts.

Learning Temporal Features: To illustrate the ability of the time-aware classifiers to track concept drift we analyzed the features used by them. In Figure 7.6 we show sports-related concepts and the days when features related to them were observed in our classifiers. We use the same set of classifiers that were used before. The intensity of the block color is proportional to the rank of that feature for the given day, with brighter color implying better ranks. We have a threshold at rank 20. Anytime a feature drops below that rank we shade it white (since it implies a less important feature). As the figure shows, our technique is able to discover both short term

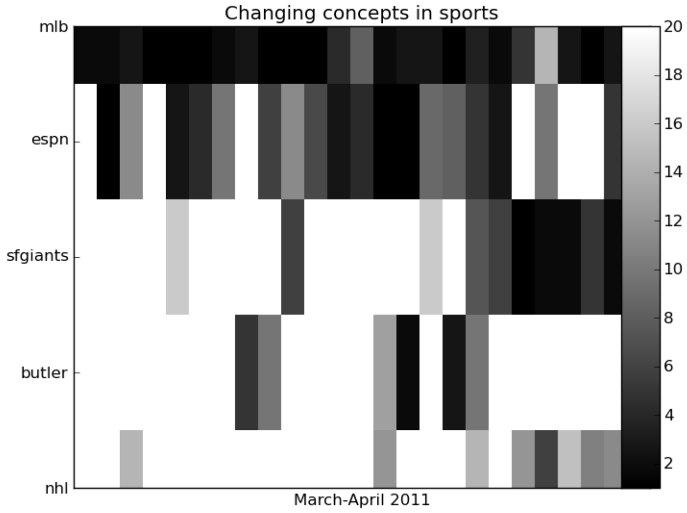


Figure 7.6: Concept discovery in May, 2010.

(Butler games) and long term (MLB) events during the time interval.

7.6 Related Work

Short text classification in the context of spam filtering has been discussed in several papers. Most of the work deals with short text in the form of mobile communications (SMS), blog comments, email summaries, etc. Cormack et.al. [16] examined several lexical feature expansion techniques for classifying short messages as spam or not. In [69] the authors model the style in which short texts are written to filter spam. They utilize features like the length of the short text and part-of-speech n-grams to build classifiers that identify spam. Other techniques like bayesian filtering [31] and the use of an external dataset [60] have also been found suitable for short text classification. In [60], the authors use an external large source of words like Wikipedia to compensate for the lack of features in short text. Though all of these papers concentrate on the problem of sparsity in short texts, there is no notion of time associated with the classifier as in our study.

Concept mining over temporal streams of data is used in areas like news classification [41] and email spam filtering [41]. One of the common approaches is the use of an ensemble of classifiers to track concept drift. In [62] the authors use an ensemble of decision tree classifiers trained on sequential data chunks. They then select appropriate classifiers depending on the data they are trying to classify. In [41] the authors develop an ensemble of classifiers to identify recurring concepts in an online stream of data for email filtering. They describe a system to sort out recurring concepts and then train the classifiers to learn these concepts. Katakis et al. in [41] develop a system which manages concept drift in news articles to provide a personalized new dissemination system. Differing from the ensemble approach they develop an instance of an incremental classifier based on naive Bayes that updates every time it receives a new news article. These techniques are generally used in domains where the data is online but not sparse.

7.7 Conclusion

In this paper we studied the problem of culling messages from time-evolving short message streams that correspond to pre-defined areas of interest. We have proposed and evaluated an expert-driven sliding window approach for classifier training in order to capture the day-to-day and hour-to-hour changes in the concepts associated with a particular category. We explored three general approaches for overcoming feature sparsity in short messages: (i) lexical-based; (ii) link-based; and (iii) collocation-based. We are encouraged by our initial results. As future work we are interested to adapt the time-sensitive classification framework to finer-grained time slices (e.g., hours, minutes) and to investigate per-account classification rather then per-message classification.

Category	**Account Name**
Technology	msnbc_tech, ForbesTech, PCMag, pcworld, cnet, CNETNews, BloombergTech, Reuters_Tech, RtrsIN_Tech, Reuters_Science, USATODAYtech, bbcscitech, bbctech, nytimesscience, WSJTech
Entertainment	TMZ, nytimesfashion, nytimesstyle, WSJLifeStyle, nytimesarts,USATODAYlife, AP_Fashion, TODAY_ent, eonline, Reuters_Entmnt, ststerling, LATshowtracker, bbccomedy, bbcentertain, nytimestheater, nytimesmusic
Politics	msnbc_politics, ReutersPolitics, PoliticalTicker, bbcpolitics, nytimespolitics, WSJPolitics, wsjindia, WSJWashington
Sports	nbc_sports, AP_NFL, RtrsIN_Sports, USATODAYsports, BBCFootball1,bbcfoot, bbcf1, MNF_on_ESPN, espn_afcsouth, espn_nfcwest, espn_afceast, espn_nfcnblog, espnafc_north, NFLLIVEonESPN, espn_afcwest, espn_nfceast, espn_nfcsouth, ESPN_MLB, MLBRumorCentral, ESPN_SEC, ESPN_Pac10, ESPN_CollegeFB, GameDayFootball, ESPNAllAmerica, ESPN_Big12, ESPN_BigTen, espn_bigeast, ESPN_ACC, CFBRumorCentral, TrueHoopNetwork, NBAonESPN, NBA_on_ESPN, ESPN_NHL, SportsNation, espn, nytimessports, WSJOlympics, WSJSports

Table 7.8: Seed Twitter accounts used for snowball sampling.

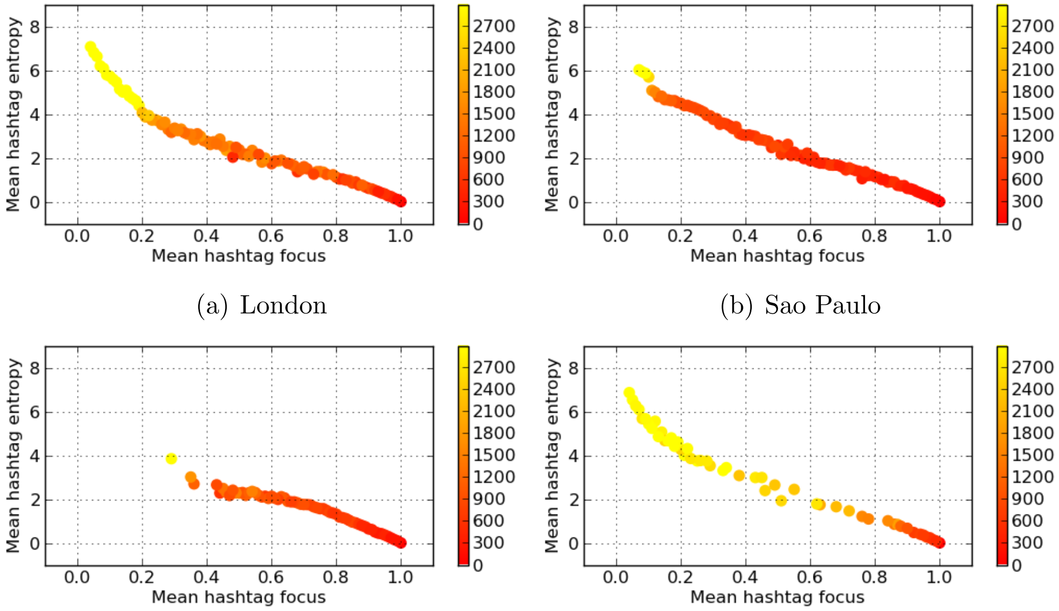
8. VISUALIZATION OF LOCATIONS USING GEO BASED SOCIAL TRAILS*

In this section, we are interested in visualizing locations social trails. In particular, we ask the following questions. How can the spatio-temporal characteristics of hashtags describe locations? Are some locations more "impactful" in terms of the hashtags that originate there? To answer these questions, we present two methods for characterizing locations based on hashtag spatial analytics. The first method uses spatial properties – entropy and focus – to determine the nature of a location from the point of hashtag propagation using location-entropy-focus-spread plots, while the second method uses hashtag adoption times to characterize a location's impact to enable hashtag propagation. Through hashtag spatial analytics, the relative impact of locations can be measured; for example while both London and Sao Paulo are home to the most total hashtags, hashtags originating in London have a global footprint, while Sao Paulo's are mostly constrained to Brazil due to inherent language and culture constraints.

8.1 Entropy-Focus-Spread Plots

In the first technique, we first assign every hashtag to its corresponding hashtag focus location. This results in every location having a set of hashtags that were focused there. Using this set of hashtags we plot the entropy versus focus for every hashtag focused on this location *plus* indicate the mean spread for every focus-entropy pair using a color gradient. To illustrate, consider the four location-based

*Part of this section is reprinted with permission from "Spatio-Temporal Dynamics of Online Memes: A Study of Geo-Tagged Tweets" by Krishna Y. Kamath, James Caverlee, Kyumin Lee and Zhiyuan Cheng, 2013. World Wide Web. 22nd. Copyright 2013 by Association for Computing Machinery (ACM).



(a) London

(b) Sao Paulo

(c) Ankara

(d) Los Angeles

Figure 8.1: (Color) Entropy-Focus-Spread plots for four cities. Local hashtags – with a high focus and a low entropy – are located in the bottom-right of each figure; global hashtags – with a low focus and a high entropy – are located in the top-left of each figure. The mean spread for every focus-entropy pair using a color gradient: high values in a lighter yellow, while lower values of spread are in red.

entropy-focus-spread plots in Figure 8.1 – one for London, Sao Paulo, Ankara, and Los Angeles. Recall that London, Sao Paulo, and Los Angeles are among the top-5 locations in terms of total hashtags, while Ankara ranks much lower.

First, we observe that locations that have high hashtag counts have a complete spectrum of hashtags on the plots. Recall that local hashtags occur on the right-bottom of such plots, while global hashtags are on the left-top. Here we see that the popular locations are the focal points (or "champions") for both local and global hashtags. Ankara, on the other hand, is the focal point for only relatively local hashtags (with high focus and low entropy).

Second, the use of spread (with high values in a lighter yellow, while lower values of spread are in red) illustrates the relative geo-spatial footprint of hashtags that have a location as its focal point. For example, although Sao Paulo has a high total number of hashtags and a high number of total locations impacted (note the hashtags with low focus and high entropy), the geospatial footprint of Sao Paulo is relatively low (note the very little yellow among these hashtags). The hashtags popular in Sao Paulo have high entropy because they are spread over several locations but all these locations are close to each other resulting in a smaller spread. Los Angeles on the other hand has a global impact; hashtags that become popular in Los Angeles tend to be popular in a larger geographical area.

8.2 Measuring Spatial Impact

The second spatial analytics technique directly evaluates the impact a location has on other locations by measuring the hashtag-based spatial impact. We define the *spatial impact* $\mathcal{I}_{l_i \to l_j}$ of location l_i on l_j as a score in the range $[-1, 1]$, such that -1 indicates l_i adopts a hashtag only after l_j has adopted it, $+1$ indicates l_j adopts a hashtag only after l_i adopts it and 0 indicates the locations are independent of each other and adopt hashtags simultaneously.

For example, consider the three cases shown in Figure 8.2. When hashtags are generated between a pair of locations as shown in (a) we want $\mathcal{I}_{l_1 \to l_2} = 1$, when as shown in (b) we want $\mathcal{I}_{l_1 \to l_2} = -1$, and when as shown in (c) we want $\mathcal{I}_{l_1 \to l_2} = 0$. Let $o_l^h(t)$ represent an occurrence of hashtag h in location l at time interval t. Then, we define the preceding operator $\prec$ over two sets of occurrences $O_{l_i}^h$ and $O_{l_j}^h$ as:

$$O_{l_i}^h \prec O_{l_j}^h = \{o_{l_i}^h(t) \mid t_i < t_j \ \forall \ (o_{l_i}^h(t_1), o_{l_j}^h(t_2)) \in O_{l_i}^h \times O_{l_j}^h\}$$

which gives a set of all occurrences of h in l_1 that precede l_2 in the cartesian product

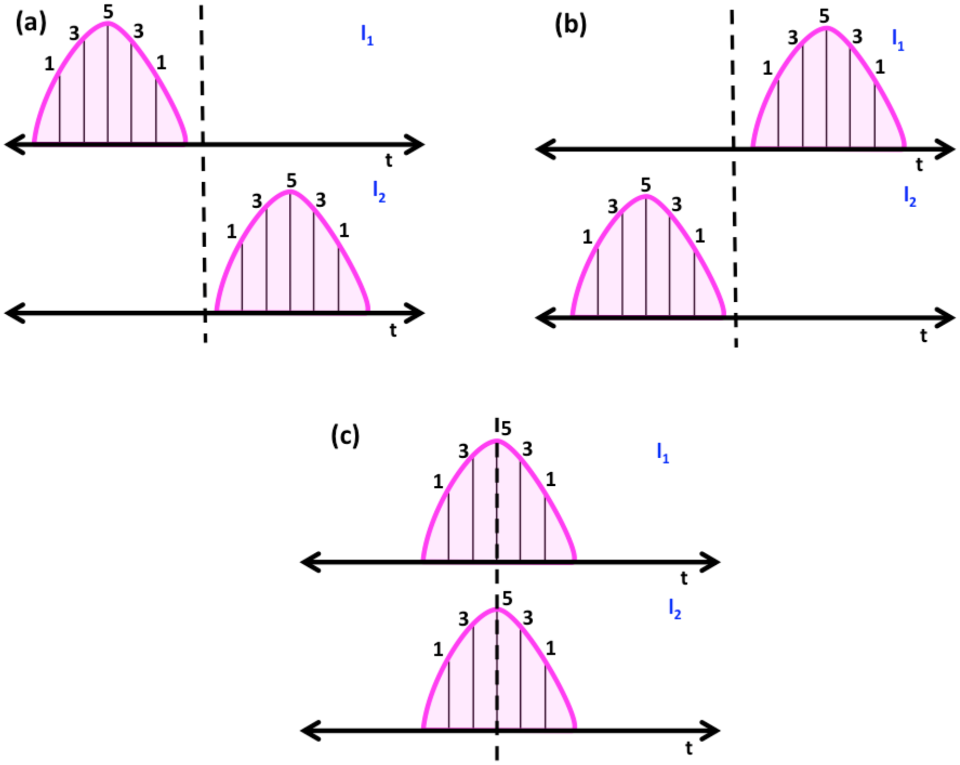


Figure 8.2: Example of hashtag adoption for two locations l_1 and l_2. In (a) l_1 adopts all of its hashtags before l_2. In (b) l_1 adopts all of its hashtags after l_2. In (c) l_1 and l_2 adopt hashtags simultaneously.

of their occurrences. Similarly, we define the succeeding operator $\succ$ as:

$$O^h_{l_i} \succ O^h_{l_j} = \{o^h_{l_i}(t) \mid t_i > t_j \;\forall\; (o^h_{l_1}(t_1), o^h_{l_j}(t_2)) \in O^h_{l_i} \times O^h_{l_j}\}$$

which gives the set of all occurrences of h in l_1 that succeed l_2 in the Cartesian product of their occurrences. We now define the spatial impact of location l_i on l_j as the average of hashtag specific spatial impact values, $\mathcal{I}^h_{l_i \to l_j}$, for all hashtags that occur in both the locations:

$$\mathcal{I}_{l_i \to l_j} = \frac{\sum_{h \in H_{l_i} \cup H_{l_j}} \mathcal{I}^h_{l_i \to l_j}}{|H_{l_i} \cup H_{l_j}|}$$

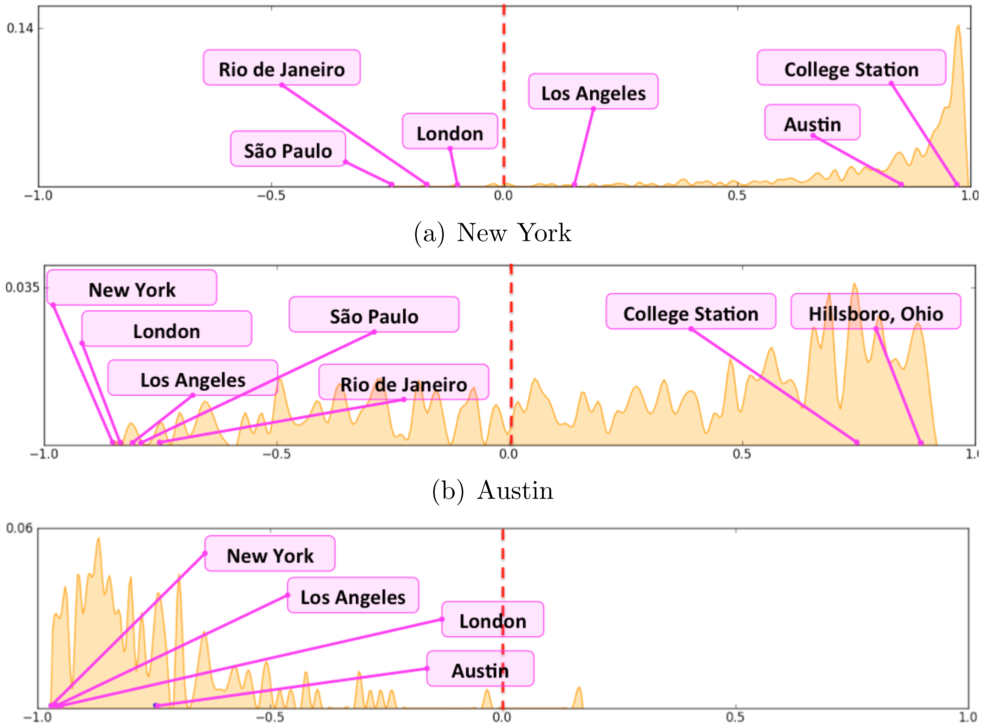
where, $\mathcal{I}^h_{l_i \to l_j}$ is defined as:

$$\mathcal{I}^h_{l_i \to l_j} = \begin{cases} \frac{|O^h_{l_i} \prec O^h_{l_j}| - |O^h_{l_i} \succ O^h_{l_j}|}{|O^h_{l_i} \times O^h_{l_j}|} & \text{if } h \in H_{l_i} \text{ and } h \in H_{l_j} \\ 1 & \text{if } h \in H_{l_i} \text{ only} \\ -1 & \text{if } h \in H_{l_j} \text{ only} \end{cases}$$

The impact is 1 if a hashtag is posted only in l_i, as l_i clearly impacts l_j in this case. For similar reasons the impact is -1 when a hashtag is posted only in l_j. To understand the case when a hashtag is observed in both locations consider the example shown in Figure 8.2. In all three cases $|O^h_{l_1}| = 13$, $|O^h_{l_2}| = 13$ and $|O^h_{l_1} \times O^h_{l_2}| = 169$.

- **Case (a)**: $|O^h_{l_1} \prec O^h_{l_2}| = 169$ and $|O^h_{l_1} \succ O^h_{l_2}| = 0$. Hence, $\mathcal{I}^h_{l_i \to l_j} = \frac{169-0}{169} = 1$.
- **Case (b)**: $|O^h_{l_1} \prec O^h_{l_2}| = 0$ and $|O^h_{l_1} \succ O^h_{l_2}| = 169$. Hence, $\mathcal{I}^h_{l_i \to l_j} = \frac{0-169}{169} = -1$.
- **Case (c)**: $|O^h_{l_1} \prec O^h_{l_2}| = 62$ and $|O^h_{l_1} \succ O^h_{l_2}| = 62$. Hence, $\mathcal{I}^h_{l_i \to l_j} = \frac{62-62}{169} = 0$.

We visualize the spatial impact of a location using a *spatial impact plot*. The x-axis represents the spatial impact values and is in the range $[-1, 1]$; the y-axis shows the distribution of locations at these values. Examples of impact plots for three locations can be found in Figure 8.3. In every impact plot, locations on the left half of the plot are *impacting* locations and the locations on the right half of the plot are *impacted* locations. Hence, plots for famous and large locations are generally right-heavy as they impact many locations. Plots for small locations are mostly left-heavy as they are impacted by many locations. For example, the impact plot for New York is right heavy since New York is an “early adopter” with a high spatial impact on other locations. Interestingly, New York is actually impacted by both Sao Paulo and Rio de Janeiro, since Portuguese hashtags tend to flow from Brazil to Portuguese-speaking neighborhoods in New York, whereas hashtags from



(a) New York

(b) Austin

(c) College Station

Figure 8.3: Spatial impact plots for three locations. Locations to the left of the origin are "early adopters" relative to the baseline location. New York has a high impact, with almost all cities to the right of its origin. College Station, on the other hand, is low impact since it only adopts hashtags after almost all other cities.

New York are less likely to flow to Brazil. College Station (home to Texas A&M) is fairly small, with a left-heavy distribution, indicating that it is a "late adopter". Austin, on the other hand, has a balanced spatial impact, being both impacted by many locations and impacting many other locations.

8.3 Summary

In this section, we proposed methods to visualize social trails using spatio-temporal properties. Using these visualizations we show tha relative impact of locations can be measured; for example while both London and Sao Paulo are home to the most total hashtags, hashtags originating in London have a global footprint, while Sao Paulo's are mostly constrained to Brazil due to inherent language and culture constraints.

9. SUMMARY AND FUTURE RESEARCH OPPURTUNITIES

In this section, we present a summary of this dissertation and potential future research avenues in this area.

9.1 Summary

In this dissertation research, we focused on the real-time social trails that reflect the digital footprints of crowds of real-time web users in response to real-world events or online phenomena. These digital footprints correspond to the artifacts strewn across the real-time web like posting of messages to Twitter or Facebook; the creation, sharing, and viewing of videos on websites like YouTube; and so on. While access to social trails could benefit many domains there is a significant research gap toward discovering, modeling, and leveraging these social trails. Hence, this dissertation made three contributions:

First, we developed a suite of efficient techniques for discovering social trails from large-scale real-time social systems. We viewed a social trail as an evolving set of *transient crowds* and focused on the task of first extracting these transient crowds. Each transient crowd (or just crowd) is a potentially short-lived ad-hoc collection of users (and their associated content) at the core of a social trail that triggers its formation and contributes to its evolution. Concretely, we first developed a communication-based method using temporal graphs for discovering social trails on a stream of conversations from social messaging systems like instant messages, emails, Twitter directed or @ messages, SMS, etc. We then developed a content-based method using locality sensitive hashing for discovering content based social trails on a stream of text messages like Tweet stream, stream of Facebook messages, YouTube comments, etc. We evaluated the performance of our social trail discovery algorithms over Twitter datasets and through extensive experimental study, we found our algorithms to be significantly efficient while maintaining high-quality crowds as compared to other approaches.

Second, we developed a framework for modeling and predicting the spatio-temporal dynamics of social trails. In particular, we developed a probabilistic model that synthesized two conflicting hypotheses about the nature of online information spread: (i) the spatial influence model, which asserts that social trails propagates to locations that are close by; and (ii) the community affinity influence model, which asserts that social trail propagates between locations that are culturally connected, even if they are distant. We tested these models in the context of the geospatial footprint of 755 million geo-tagged hashtags and found that while the spatial influence model had a higher impact than the community affinity influence model in predicting the spread, its combination with community affinity influence model gave the best performance, suggesting that both distance and community are key contributors to social media spread. The combination of these models was able to predict flow close to 80% accuracy of the best possible model.

Third, we developed a set methods for social trail analytics and leveraging social trails for prognostic applications like real-time content recommendation, personalized advertising, and so on. We first analyzed geo-spatial social trails of hashtags from Twitter and investigate their spatio-temporal dynamics. Based on the insights we gained during modeling of social trails and and the analysis of their geo-spatial properties we addressed the challenge of classifying social trails efficiently on real-time social systems. We proposed an expert-driven framework for time-aware topical classification framework for social trails. We showed how expert streams may seed classification, and proposed a sliding-window training approach for adaptive topical classification. Additionally, we explored techniques for augmenting short messages using feature-based, link-based and collocation expansion. Through experimental study over Twitter, we found good performance of the proposed method for ongoing expert-driven topical classification of social trails.

9.2 Future Research Opportunities

Over the past few years we have seen rapid adoption of social media, predominantly as a platform to generate, share and consume information. In the coming years, we believe social media will move from just being a platform for communication to a framework on top of which Internet applications will be built. We are already seeing example of this in areas like Social Commerce (Chirpify, Amex-Twitter sync), Social TV (Audience feedback into TV programs, real-time polling), real-time advertising (ex. around super-bowl events) and so on. These services will not only motivate consumers to use social media more ensuring its continued growth, but it will also demand development of new computational approaches for monitoring, analyzing, and distilling information from the prospective web of real-time content.

- **Geo-Based Crowds**: In Section 2 and Section 3, we discussed approaches to discover crowds. There we looked at crowds from two perspectives: (i) Crowds based on communication between users; and (ii) Crowds based on the content of the messages. Future research in this direction can look at other perspectives to discover crowds like location. These geo-based crowds could be discovered based on the geo co-ordinates used in the messages (tweets), and the crowds could be further broken down based on the content in the messages and communication graph. This gives a location specific content and user communication based crowds. For example, this method can be used to discover crowds that are local to New York and related to Occupy Wall Street.
- **Geo-Spatial Analysis Coupled With Social Networks**: In Section 5 and

Section 8, we described methods to analyze and model social trails. In this we focussed mostly on the geo-spatial aspects of social trail propagation. We believe future research in this direction can move beyond geo-spatial and look at the underly social network and the impact this network has on the propagation of social trails. Social networks are are very important when it comes to information propagation and there has been plenty of research related to analyzing and modeling information propagation on them. We believe combining the models related to geo-spatial propagation like spatial and community, with the models of information propagation on social networks can result in models that better simulate social trail propagation.

REFERENCES

[1] Charu C. Aggarwal. A framework for clustering massive text and categorical data streams. In *In: Proc. SIAM Conference on Data Mining*, pages 477–481, 2006.

[2] Charu C. Aggarwal, Jiawei Han, Jianyong Wang, and Philip S. Yu. A framework for clustering evolving data streams. In *Proceedings of the 29th International Conference on Very Large Data Bases - Volume 29*, VLDB '2003, pages 81–92. VLDB Endowment, 2003.

[3] Sitaram Asur, Srinivasan Parthasarathy, and Duygu Ucar. An event-based framework for characterizing the evolutionary behavior of interaction graphs. In *KDD '07*, pages 913–921, New York, NY, USA, 2007. ACM.

[4] L. Backstrom, J. Kleinberg, R. Kumar, and J. Novak. Spatial variation in search engine queries. In *Proceeding of the 17th International Conference on World Wide Web*, pages 357–366. ACM, 2008.

[5] L. Backstrom, E. Sun, and C. Marlow. Find me if you can: improving geographical prediction with social and spatial proximity. In *Proceedings of the 19th International Conference on World Wide Web*, pages 61–70. ACM, 2010.

[6] Lars Backstrom, Dan Huttenlocher, Jon Kleinberg, and Xiangyang Lan. Group formation in large social networks: membership, growth, and evolution. In *KDD '06*, pages 44–54, New York, NY, USA, 2006. ACM.

[7] C. Bauckhage. Insights into internet memes. *Proc. ICWSM2011*, pages 42–49, 2011.

[8] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[9] A. Brodersen, S. Scellato, and M. Wattenhofer. Youtube around the world: geographic popularity of videos. In *Proceedings of the 21st International Conference on World Wide Web*, pages 241–250. ACM, 2012.

[10] Frances Cairncross. *The Death of Distance: How the Communications Revolution is Changing Our Lives.* Harvard Business School Press, Boston, 2001.

[11] Moses S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, STOC '02, pages 380–388, New York, NY, USA, 2002. ACM.

[12] Z. Cheng, J. Caverlee, and K. Lee. You are where you tweet: a content-based approach to geo-locating twitter users. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, pages 759–768. ACM, 2010.

[13] Z. Cheng, J. Caverlee, K. Lee, and D.Z. Sui. Exploring millions of footprints in location sharing services. *AAAI ICWSM*, 2011.

[14] H.L. Chieu and H.T. Ng. Named entity recognition: a maximum entropy approach using global information. In *Proceedings of the 19th International Conference on Computational Linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics, 2002.

[15] Hyunyoung Choi and Hal Varian. Predicting the present with google trends. Technical report, Google, http://www.google.com/googleblogs/pdfs/google_predicting_the_present.pdf, 2009.

[16] Gordon V. Cormack, José María Gómez Hidalgo, and Enrique Puertas Sánz. Spam filtering for short messages. In *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management*, CIKM '07, pages 313–320, New York, NY, USA, 2007. ACM.

[17] E. Cunha, G. Magno, G. Comarela, V. Almeida, M.A. Gonçalves, and F. Benevenuto. Analyzing the dynamic evolution of hashtags on twitter: a language-based approach. In *Proceedings of the Workshop on Language in Social Media (LSM 2011)*, pages 58–65, 2011.

[18] Douglass R. Cutting, David R. Karger, Jan O. Pedersen, and John W. Tukey. Scatter/gather: a cluster-based approach to browsing large document collections. In *Proceedings of the 15th ACM SIGIR*, SIGIR '92, pages 318–329, New York, NY, USA, 1992. ACM.

[19] N. Dalvi, R. Kumar, and B. Pang. Object matching in tweets with spatial models. In *Proceedings of the fifth ACM International Conference on Web Search and Data Mining*, pages 43–52. ACM, 2012.

[20] D. Davidov, O. Tsur, and A. Rappoport. Enhanced sentiment learning using twitter hashtags and smileys. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 241–249. Association for Computational Linguistics, 2010.

[21] Inderjit Dhillon, Yuqiang Guan, and Brian Kulis. A fast kernel-based multilevel algorithm for graph clustering. In *KDD '05*, pages 629–634, New York, NY, USA, 2005. ACM.

[22] J. Ding, L. Gravano, and N. Shivakumar. Computing geographical scopes of web resources. In *Proceedings of the 26th International Conference on Very Large Data Bases*, VLDB '2000. Morgan Kaufmann Publishers Inc., 2000.

[23] T. Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74, 1993.

[24] Twitter Engineering. 200 million tweets per day. http://blog.twitter.com/2011/06/200-million-tweets-per-day.html, June 2011.

[25] Facebook. Facebook - fact sheet. http://newsroom.fb.com/Key-Facts, 2012.

[26] Wei Fan, Yusuke Koyanagi, Koichi Asakura, and Toyohide Watanabe. Clustering over evolving data streams based on online recent-biased approximation. In Debbie Richards and Byeong-Ho Kang, editors, *Knowledge Acquisition: Approaches, Algorithms and Applications*, volume 5465 of *Lecture Notes in Computer Science*, pages 12–26. Springer Berlin Heidelberg, 2009.

[27] Gary W. Flake, Robert E. Tarjan, and Kostas Tsioutsiouliklis. Graph clustering and minimum cut trees. *Internet Mathematics*, 1(4):385–408, 2004.

[28] Foursquare. About foursquare. https://foursquare.com/about/, April 2013.

[29] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Similarity search in high dimensions via hashing. In *Proceedings of the 25th International Conference on Very Large Data Bases*, VLDB '99, pages 518–529, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[30] A V Goldberg and R E Tarjan. A new approach to the maximum flow problem. In *STOC '86*, pages 136–146, New York, NY, USA, 1986. ACM.

[31] J.M. Gómez Hidalgo, G.C. Bringas, E.P. Sánz, and F.C. García. Content based sms spam filtering. In *Proceedings of the 2006 ACM Symposium on Document Engineering*, pages 107–114. ACM, 2006.

[32] M. Gomez-Rodriguez, J. Leskovec, and A. Krause. Inferring networks of diffusion and influence. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5(4):21, 2012.

[33] R. E. Gomory and T. C. Hu. Multi-terminal network flows. *Journal of the Society for Industrial and Applied Mathematics*, 9(4):551–570, 1961.

[34] Linghui Gong, Jianping Zeng, and Shiyong Zhang. Text stream clustering algorithm based on adaptive feature selection. *Expert Syst. Appl.*, 38:1393–1399, March 2011.

[35] Google. Explore flu trends around the world. http://www.google.org/flutrends/, January 2013.

[36] D.J. Hand and R.J. Till. A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine Learning*, 45(2):171–186, 2001.

[37] Bernardo A. Huberman, Daniel M. Romero, and Fang Wu. Social Networks that Matter: Twitter Under the Microscope. *Social Science Research Network Working Paper Series*, December 2008.

[38] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the Thirteenth Annual ACM Symposium on Theory of Computing*, STOC '98, pages 604–613, New York, NY, USA, 1998. ACM.

[39] Krishna Y. Kamath, James Caverlee, Zhiyuan Cheng, and Daniel Z. Sui. Spatial influence vs. community influence: modeling the global spread of social media. In *Proceedings of the 21st ACM International Conference on Information and*

Knowledge Management, CIKM '12, pages 962–971, New York, NY, USA, 2012. ACM.

[40] Krishna Yeshwanth Kamath and James Caverlee. Transient crowd discovery on the real-time social web. In *Proceedings of the fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 585–594, New York, NY, USA, 2011. ACM.

[41] I. Katakis, G. Tsoumakas, E. Banos, N. Bassiliades, and I. Vlahavas. An adaptive personalized news dissemination system. *Journal of Intelligent Information Systems*, 32(2):191–212, 2009.

[42] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 137–146. ACM, 2003.

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

[44] G. Kossinets, J. Kleinberg, and D. Watts. The structure of information pathways in a social communication network. In *Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 435–443. ACM, 2008.

[45] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web*, pages 591–600. ACM, 2010.

[46] K. Lerman and R. Ghosh. Information contagion: An empirical study of the spread of news on digg and twitter social networks. In *Proceedings of 4th Inter-*

national Conference on Weblogs and Social Media (ICWSM), 2010.

[47] J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 497–506. ACM, 2009.

[48] J. Lin, R. Snow, and W. Morgan. Smoothing techniques for adaptive online language models: topic tracking in tweet streams. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 422–429. ACM, 2011.

[49] Yu-Bao Liu, Jia-Rong Cai, Jian Yin, and Ada Wai-Chee Fu. Clustering text data streams. *Journal of Computer Science and Technology*, 23(1):112–128, 2008.

[50] E. Loper and S. Bird. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics-Volume 1*, pages 63–70. Association for Computational Linguistics, 2002.

[51] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. *Introduction to Information Retrieval.* Cambridge University Press, New York, NY, USA, 2008.

[52] Christopher D. Manning and Hinrich Schuetze. *Foundations of Statistical Natural Language Processing.* The MIT Press, 1 edition, 1999.

[53] F. Moerchen, K. Brinker, and C. Neubauer. Any-time clustering of high frequency news streams. In *Proc. Data Mining Case Studies Workshop, KDD*, 2007.

[54] M. E. J. Newman. Fast algorithm for detecting community structure in networks. http://arxiv.org/abs/cond-mat/0309508, September 2003.

[55] K. Nigam, J. Lafferty, and A. McCallum. Using maximum entropy for text classification. In *IJCAI-99 workshop on machine Learning for Information Filtering*, volume 1, pages 61–67, 1999.

[56] A. Noulas, S. Scellato, C. Mascolo, and M. Pontil. An empirical study of geographic user activity patterns in foursquare. *ICWSM'11*, 2011.

[57] Gary M Olson and Judith S Olson. Distance matters. *Human-computer Interaction*, 15(2):139–178, 2000.

[58] Alexei Oreskovic. Huffington post: Youtube video views hit 4 billion per day. http://www.huffingtonpost.com/2012/01/23/youtube-video-views_n_1223070.html, January 2012.

[59] Saša Petrović, Miles Osborne, and Victor Lavrenko. Streaming first story detection with application to twitter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 181–189, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[60] X.H. Phan, L.M. Nguyen, and S. Horiguchi. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th International Conference on World Wide Web*, pages 91–100. ACM, 2008.

[61] Michael E Porter. Location, competition, and economic development: Local clusters in a global economy. *Economic Development Quarterly*, 14(1):15–34, 2000.

[62] Sasthakumar Ramamurthy and Raj Bhatnagar. Tracking recurrent concept drift in streaming data using ensemble classifiers. In *ICMLA '07: Proceedings of the Sixth ICML*, pages 404–409, Washington, DC, USA, 2007. IEEE Computer Society.

[63] Deepak Ravichandran, Patrick Pantel, and Eduard Hovy. Randomized algorithms and nlp: using locality sensitive hash function for high speed noun clustering. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 622–629, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

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

[65] Barna Saha and Pabitra Mitra. Dynamic algorithm for graph clustering using minimum cut tree. In *ICDMW '06*, pages 667–671, Washington, DC, USA, 2006. IEEE Computer Society.

[66] J. Sankaranarayanan, H. Samet, B.E. Teitler, M.D. Lieberman, and J. Sperling. Twitterstand: news in tweets. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 42–51. ACM, 2009.

[67] S. Scellato, A. Noulas, R. Lambiotte, and C. Mascolo. Socio-spatial properties of online location-based social networks. *Proceedings of ICWSM*, 11:329–336, 2011.

[68] Jan Aart Scholte. *Globalization: A Critical Introduction*. Palgrave Macmillan, 2005.

[69] D.N. Sohn, J.T. Lee, and H.C. Rim. The contribution of stylistic information to content-based mobile spam filtering. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 321–324. Association for Computational Linguistics, 2009.

[70] Bharath Sriram, Dave Fuhry, Engin Demir, Hakan Ferhatosmanoglu, and Murat Demirbas. Short text classification in twitter to improve information filtering. In *SIGIR '10*, pages 841–842, New York, NY, USA, 2010. ACM.

[71] J. Sun, C. Faloutsos, S. Papadimitriou, and P. S. Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *13th ACM SIGKDD' 07*, pages 687–696, New York, NY, USA, 2007. ACM.

[72] W. Tobler. A computer movie simulating urban growth in the detroit region. *Economic Geography*, 46(2), 1970.

[73] O. Tsur and A. Rappoport. What's in a hashtag?: content based prediction of the spread of ideas in microblogging communities. In *Proceedings of the fifth ACM International Conference on Web Search and Data Mining*, pages 643–652. ACM, 2012.

[74] Stijn Van Dongen. Graph clustering via a discrete uncoupling process. *SIAM J. Matrix Anal. Appl.*, 30(1):121–141, 2008.

[75] Wikipedia. Universal transverse mercator coordinate system. http://en.wikipedia.org/wiki/Universal_Transverse_Mercator_coordinate_system, November 2012.

[76] Wikipedia. Infinite coordinate space. wikipedia: Examples of vector spaces, January 2013.

[77] S. Wu, J.M. Hofman, W.A. Mason, and D.J. Watts. Who says what to whom on twitter. In *Proceedings of the 20th International Conference on World Wide Web*, pages 705–714. ACM, 2011.

[78] J. Yang and J. Leskovec. Modeling information diffusion in implicit networks. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 599–608. IEEE, 2010.

[79] Surender Reddy Yerva, Zoltán Miklós, and Karl Aberer. It was easy, when apples and blackberries were only fruits. In *Third Web People Search Evaluation Forum (WePS-3), CLEF*. Citeseer, 2010.

[80] Yunyue Zhu and Dennis Shasha. Statstream: statistical monitoring of thousands of data streams in real time. In *Proceedings of the 28th International Conference on Very Large Data Bases*, VLDB '02, pages 358–369. VLDB Endowment, 2002.