GeoScope: Online Detection of Geo-Correlated Information Trends in Social Networks

Ceren Budak
cbudak@microsoft.com
Microsoft Research

Theodore Georgiou Divyakant Agrawal Amr El Abbadi
{teogeorgiou, agrawal, amr}@cs.ucsb.edu
University of California, Santa Barbara
Department of Computer Science

ABSTRACT

The First Law of Geography states "Everything is related to everything else, but near things are more related than distant things". This spatial significance has implications in various applications, trend detection being one of them. In this paper we propose a new algorithmic tool, *GeoScope*, to detect geo-trends. *GeoScope* is a data streams solution that detects correlations between topics and locations in a sliding window, in addition to analyzing topics and locations independently. *GeoScope* offers theoretical guarantees for detecting all trending correlated pairs while requiring only sublinear space and running time. We perform various human validation tasks to demonstrate the value of *GeoScope*. The results show that human judges prefer *GeoScope* to the best performing baseline solution 4:1 in terms of the geographical significance of the presented information. As the Twitter analysis demonstrates, *GeoScope* successfully filters out topics without geo-intent and detects various local interests such as emergency events, political demonstrations or cultural events. Experiments on Twitter show that *GeoScope* has perfect recall and near-perfect precision.

1. INTRODUCTION

Geography plays an important role in various aspects of our lives. As the first law of geography states "Everything is related to everything else, but near things are more related than distant things" [33]. With the advent of the Web and later online social networks, the "virtual" distance between Web users has dramatically decreased. Yet, research shows that geographical locality still matters in our choice of friends [35], our use of language and sentiment [26], as well as topical interests [14].

Geographical signals can also be used to extract relevant information from the public for crisis management [20]. Therefore, it is critical to develop social network analysis tools that have a geographical focus. Most research in this area is restricted to offline geographical measurements [5, 26]. Recently, there has been effort in online analysis of geo-trends [20, 31]. However, these works focus on defining frameworks in which data is simply geographically categorized while the task of discovering geo-intent by considering the correlation between locations and topics is not addressed. Given the large scale of data shared through online social networks, there is need for algorithmic solutions that capture geo-intent and detect informational trends in a scalable fashion. Our goal in this paper is to provide such an algorithmic tool that uses sublinear space and running time with approximation guarantees.

We aim to detect trends of true geographical nature rather than simply identifying frequent elements in various locations. Global trends, which incidentally would be trending in various locations, carry no geographical significance and are irrelevant from the perspective of our study. In order to distinguish from such topics, we focus on the challenging problem of identifying correlations of information items with different geographical places in an efficient manner.

We propose *GeoScope*; an algorithmic tool for detecting geo-trends in online social networks by reporting *trending* and *correlated* location - topic pairs. *GeoScope* also captures the temporality of trends by detecting geo-trends along a sliding window. The use of different window sizes can also allow trends of different time granularity to be detected. To the best of our knowledge, this is the first work that detects spatial information trends in social networks by capturing correlations in a multi-dimensional data stream. *GeoScope* has provable accuracy guarantees even though it requires sublinear memory and amortized running time. Such a scalable algorithmic tool can be used in real large-scale social networks to provide reliable and *online* detection of local interests or even crisis events. Our analysis on a Twitter data set shows that *GeoScope* detects significant events ranging from emergency events to political demonstrations to concerts and sports events. The fast detection of emergency events such as the March 11 Japan earthquake indicate the possible value of *GeoScope* in crisis management [30].

In §2, we start by summarizing related work. In §3 we introduce the characteristics that an ideal geo-trend detection tool should have and show that an exact on-line solution is not scalable. In §4, we propose an approximate solution, called *GeoScope*, and provide proofs of its accuracy and efficiency. Next, *GeoScope* is experimentally evaluated in §5. Finally, §6 concludes the paper.

2. RELATED WORK

Here we provide an overview of studies in social networks and data streams research that relate to the problem studied in our paper.

Social Networks Analysis: Trend detection in social networks has been an important research area in the recent years [13, 17, 19, 2]. Kwak et al. [17] study trending topics reported by Twitter and compare them with trends in other media, showing that the majority of topics are headlines or persistent news. In [19] Leskovec et al. study temporal properties of information by tracking "memes" across the blogosphere. Teitler et al. [31] collect, analyze, and display news stories shared in Twitter on a map interface. Hong et al. [14] focus on user profiling from a geographical perspective by

Proceedings of the VLDB Endowment, Vol. 7, No. 4

modeling topical interests through geo-tagged messages in Twitter. MacEachren et al. [20] aim to identify significant events in different localities for crisis management. Another framework for detecting spatiotemporal topics has been introduced in the context of online blogs [7]. These studies provide high level frameworks while we provide an efficient algorithmic tool with accuracy guarantees.

Geo-tagging social content is an important sub-task in our study [5, 36, 9]. Cheng et al. [5] develop a probabilistic framework to identify city-level user locations based on tweet content. However, this solution requires a large number of tweets per person for high accuracy. It is also a batch process while our goal is to detect trends in a streaming fashion requiring geo-tagging to be performed in an ad-hoc manner. Other studies [36, 9] give accurate aggregate results but have low performance per user. Instead of such Bayesian models with large error margins, we use simple methods to extract place names from tweets and user profiles.

Data Streams and Detecting Significant Co-occurences: Algorithms for answering frequent elements queries (heavy-hitters) are broadly divided into two categories: *sketch-based* and *counter-based*. In the *sketch-based* techniques [4, 6, 16], the entire data stream is represented as a summary *sketch* which is updated as the elements are processed. The *counter-based* techniques [24, 21, 8] monitor a subset of the stream elements and maintain an approximate frequency count. Although our method relies on frequent element detection, it necessitates frequent element detection in the multidimensional space. Therefore, we cannot directly apply approaches developed in the context of heavy-hitters.

There has been some effort in multi-dimensional data streams [18, 25, 37]. The closest study to our paper is presented in [25] which addresses the problem of fraud detection in Internet advertising. The proposed solution, *SLEUTH*, models single-publisher attack discovery as a problem of finding correlations in multidimensional data consisting of two dimensions; the publisher, and the IP address of a machine. They detect *correlated publisher-IP pairs* to detect fraudulent behavior. *SLEUTH* does not support deletions and therefore extracts correlated items from the *entire* data stream. Unlike SLEUTH [25], *GeoScope* is a sketch based solution that allows for a sliding window implementation. Moreover, *SLEUTH* makes the assumption that the traffic characteristics of non-fraudulent publishers and IPs are stable. Such an assumption is not applicable in online social networks where trends are highly temporal. Lappas et al. [18] study burstiness in documents in a spatiotemporal manner. While their methodology also captures the notion of geography and time, it focuses on data burstiness and not geo-intent. In [37], the authors consider correlations between time series in a sliding window and solve this problem through Discrete Fourier Transforms and a three level time interval hierarchy. This study detects correlations between time series rather than data items.

Identifying significant co-occurrences is also an important problem for the data mining and information retrieval communities [27, 37, 22]. In [27], the authors detect emerging concepts in textual data mining and identify related concepts through a co-occurrence computation. This task is performed through an expensive full matrix computation, making this technique impractical to apply on online trend detection problems. [22] assesses the co-occurrences of keywords in recent tweets to group them together. For this purpose, a small list of recent tweets is retrieved for each bursty keyword and keywords that are found to co-occur in a relatively large number of recent tweets are placed in the same group. Unfortunately, the authors do not provide the details of their algorithm. BlogPulse [12] is another service that discovers trends from blogs on a daily basis. It combines a number of statistical techniques for finding trending phrases based on their frequency of appearance in relation to other phrases. Then, a clustering technique is applied on the trending phrases to build clusters that are characterized by phrases that frequently co-occur in blogs. Note that, similar to [27], this study presents an expensive solution based on full matrix computations while our goal is to approximately solve this problem. [1] considers the correlation between two items as their mutual information. The authors use a random sample of relevant documents along with the precomputed IDF values to approximate the mutual information for two keywords. Note that the precomputation of IDF values eliminate the possibility of a streaming solution.

3. DETECTING GEO-TRENDS

In this section, our goal is to identify the characteristics that comprise a useful geo-trend detection tool. We aim to define which locations, topics and correlations are necessary and sufficient to provide a rich geo-trend detection tool. These characteristics lead to the three main premises of our algorithmic design.

We denote the set of all topics as $T = \{t_1, t_2, ..., \}$ and the set of all locations as $L = \{l_1, l_2, ...\}$; $|T|$ and $|L|$ denote the number of topics and locations. We refer to the stream of location-topic *pairs* of the form (l_i, t_j) as S. Note that various social networks can easily be mapped to this generic framework. By introducing *GeoScope* under such a framework, we aim to provide a trend detection tool that has wide applicability. In the following sections we will assume that the number of distinct topics and locations are known in advance and do not change. However, our solution also works for such cases by simply creating larger sketches as the data range grows [16].

3.1 Geo-Trend Basics

A basic geo-trend detection tool should provide a high level overview of the popularity of *locations* and *topics*. Such a tool should answer queries such as *"What fraction of the mentions in the current time window are about topic t_x (or from location l_i)?"* efficiently and accurately. This notion can be formalized by the following premise:

PREMISE 1. *The frequency of any topic t_x and any location l_i in the current time window should be reported in an accurate and timely fashion.*

As Premise 1 states, we focus on detecting trends in the current time window; therefore the solutions that will be introduced from now on are based on a sliding window notion. The size, either defined as a maximum number of (l_i, t_x) to keep track of or a time period such as a day or an hour, is set by the user. This premise ensures tracking global trends in the social network. Not only can one identify the interesting topics but also keep track of most active geographical locations in the network. This task can be achieved by traditional heavy hitters approaches and has already been addressed to a large extent in recent research. In this paper, we aim to reach beyond that and identify *geo-trends* that provide the link between the topics and locations by capturing the correlation between the two. Consider a stream consisting of pairs (l_i, t_x) where l_i is the *geo-origin* of a tweet and t_x is the topic of the tweet. In this context, geo-trends can be captured through the following premise:

PREMISE 2. *All* significantly correlated *location-topic pairs in the current time window can be retrieved at any particular time in an efficient and accurate manner. A location-topic pair (l_i, t_x) is* significantly correlated *if at least ϕ fraction of all mentions* from *location l_i are about topic t_x and at least ψ fraction of all mentions about topic t_x are* from *location l_i.*

Here ϕ captures the *dominance* of topic t_x in location l_i and ψ captures the *support* of location l_i for topic t_x. Assume the following

list of location-topic pairs: $\{(l_1,t_1), (l_2,t_1), (l_3,t_1), (l_1,t_2), (l_1,t_3), (l_2,t_3), (l_2,t_3)\}$ and that $\phi = \psi = 0.5$. The only correlation reported based on Premise 2 is (l_2,t_3). Correlation (l_1,t_2) is not reported even though l_1 has enough *support* for t_2 since t_2 is not *dominant* in l_1. A similar filtering can be observed for the correlation (l_3,t_1) since t_1 is a global trend, appearing equally in all three locations and hence, l_3 lacks *support* for t_1. Through this premise, the geo-trend detection tool captures the interests in different localities and provides means for serving important applications such as crisis management.

We rely on the *dominance* (ϕ) and *support* (ψ) parameters rather than just *statistically significant* correlations. Statistical analysis can compute the association strength between a pair of location and topic by comparing their *expected* and *observed* frequencies. The χ^2 statistic is a classical method commonly used for this type of analysis. While, the notion of statistical significance [11] is an interesting concept, the application of statistical methods such as χ^2 test would reject most null hypotheses, i.e. a location-topic pair not being correlated, due to the large sample size [3]. This would result in unmanagable or even meaningless correlations being detected. Therefore, we believe leaving the choice of ϕ and ψ to be determined based on the specific application is more practical and useful compared to the detection of statistically significantly correlated location-topic pairs.

One other important characteristic of a useful trend detection tool is its ability to filter out *insignificant* information. Given the large number of locations and topics and their Zipfian distribution of popularity, a scalable and useful trend detection tool should also filter out *unpopular* correlations. Consider a hypothetical location l_i consisting of only one user who is interested in a highly uncommon topic t_x. If there are no restrictions on the significance of locations, the pair (l_i,t_x) would be reported as a correlated pair. Given the Zipfian nature of popularity of locations and topics, it is easy to see that the list of correlations involving such locations would grow large. In order to avoid reporting an unmanageably large list of location-topic correlations, there should be a lower bound on the importance of a given location for it to be reported by the geo-trend detection tool. This leads us to the final premise of our algorithm:

PREMISE 3. *Geo-trend detection should identify a list of "all" and "only" the locations that are at least θ-frequent in the current time window and limit the reported correlations to such locations.*

A θ-frequent location in a window of N elements is a location that occurs at least θN times where $0 \leq \theta \leq 1$. Through this premise, geo-trend detection is guaranteed to capture significant locations while also keeping the number of reported locations at a manageable size. Such a requirement also filters out locations for which there is not enough data to infer any geographical interest. Given that Premise 2 dictates a correlation to be reported *only* if *both* the location and the topic are heavy-hitters for each other, Premise 3 also ignores unpopular topics by eliminating unpopular locations. This is because unpopular topics cannot be frequent for popular locations; the only locations tracked for correlations.

So far we defined *geo-trends* where locations represent the *geo-origin* of information. A similar definition could be constructed to detect the correlations in a stream of pairs (l_y,t_y) where l_j is the *geo-focus* of the shared content and t_y is the topic. In this case, the location-topic pair (l_y,t_y) is significantly correlated if at least ϕ fraction of all mentions *about* location l_i are also about topic t_x and at least ψ fraction of all mentions about topic t_x are *about* location l_i. Again, the correlations are filtered due to Premise 3, meaning no correlation whose geo-focus is a location with less than $\theta * N$ occurrences in a window of N elements gets reported.

Next, we will provide the formal problem definition that addresses the three premises introduced above.

3.2 Problem Definition

Given a stream S of location-topic *pairs* of the form (l_i,t_j) and three user defined frequency thresholds θ, ϕ, and ψ in the interval [0, 1]; our goal is to keep track of **(i)** the frequencies $F(l_i)$ ($F(t_x)$) of all locations l_i (topics t_x) and **(ii)** all pairs (l_i,t_x) s.t. $F(l_i) > \lceil \theta N \rceil$, $F(l_i,t_x) > \lceil \phi F(l_i) \rceil$, and $F(l_i,t_x) > \lceil \psi F(t_x) \rceil$ in the current time window. Here $F(l_i,t_x)$ is the number of *pairs* on topic t_x from location l_i; $F(l_i)$ is the number of the *pairs* from l_i in the current time window; and $F(t_x)$ is the number of *pairs* on t_x. The window size can be set in terms of the number of elements or an actual time window. In the former case, the number of elements N in the current window is defined by the user. Since the frequency of each topic and location is tracked, Premise 1 is satisfied. As all the correlated pairs are determined, Premise 2 is captured by definition. Finally, by setting $F(l_i) > \lceil \theta N \rceil$ we ensure Premise 3.

3.3 Exact Solution

An exact solution that solves the problem described in Section 3.2 requires keeping track of all possible pairs in a given window. We will prove this statement, by focusing on Premise 2 alone. The full solution that also satisfies Premises 3 and 1 is at least as hard.

THEOREM 3.1. *Any exact solution for the problem of detecting geo-correlated trends in a sliding window requires keeping exact and complete information about all pairs in the given window.*

PROOF. Given a stream $S = \{...,t_{i+1},t_{i+2},...,t_{i+m},...\}$ and a window size m, construct a 2-dimensional stream as follows, $S' = \{...,(l_1,t_{i+1}),(l_1,t_{i+2}),...,(l_1,t_{i+m}),...\}$, by appending some location l_1 as the first value of every pair. An answer to the query about correlations at time step $i+m$ in the constructed stream with thresholds ϕ and $\psi = 1 - \frac{1}{m}$ and $\theta = 1$ can be directly translated into an answer to a query about frequent elements in the original stream with threshold ϕ. Therefore, answering the correlated geo-trend query in S' is equivalent to answering the frequent elements query in S which requires complete information about all elements. □

Given Theorem 3.1, the large number of distinct topics and locations that are usually encountered on online social networks create significant challenges. For instance as we later discuss in Section 5.1, there are over 50K cities and over 2.3M unique topics in our dataset which results in over 115 billion different possible pairings. The *rate* at which information is shared introduces yet another challenge. For instance, there are on average 400 million tweets shared on Twitter per day [34]. Due to such challenges, the exact solution of keeping track of all possible pairs of locations and topics becomes infeasible. To address this we propose an approximation method with sub-linear memory and processing requirements.

4. GeoScope

Given the infeasibility of the exact solution, we now propose *GeoScope* that requires sublinear memory and amortized running time while still providing accuracy guarantees. The main idea behind *GeoScope* is to limit the number of monitored locations by tracking those that are at least θ-frequent and to further limit the number of monitored topics by tracking a topic t_x *only if* t_x is ϕ-frequent for at least one location and then only track ψ-frequent locations for each such topic. Given that there can be at most $\lceil \frac{1}{\theta} \rceil$ θ-frequent locations at a given time, each of which can have up to $\lceil \frac{1}{\phi} \rceil$ topics that are ϕ-frequent, the number of elements to track can be bounded by a small number. As we will demonstrate later

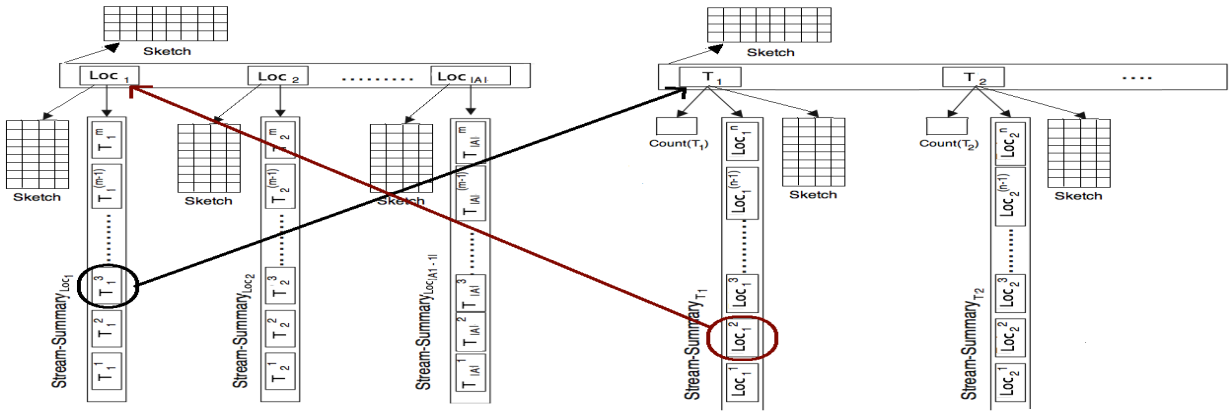


Figure 1: Overview of *GeoScope* Data Structures: *Location-StreamSummary-Table* (on the left) keeps track of ϕ-frequent topics for θ-frequent locations. *Topic-StreamSummary-Table* (on the right) keeps track of ψ-frequent locations for each topic that is ϕ-frequent for at least one location. Here the third most important topic for Loc_1 is T_1 and the second most important location for T_1 is Loc_1

on, in order to provide accuracy guarantees, *GeoScope* relaxes the number of locations to track from $\lceil \frac{1}{\phi} \rceil$ to $\lceil \frac{1}{\phi-\varepsilon} \rceil$ where $\varepsilon \ll \theta$.

4.1 GeoScope Data Structures

An overview of *GeoScope* 's structure is provided in Figure 1. In this section we briefly describe *GeoScope* and its subcomponents. As it can be seen in Figure 1, *GeoScope* consists of two main components. The first component, *Location-StreamSummary-Table*, consists of a hashtable and a sketch structure. The sketch structure is required to enable deletions, something necessary for a sliding window solution. It keeps track of frequencies of locations by allowing both insertion and deletion operations and guarantees that the real frequencies are never underestimated [16]. The second subcomponent of *Location-StreamSummary-Table* is a hashtable that contains a $StreamSummary_{l_i}$ structure for each location l_i that has a current estimated relative-frequency of at least θ. Given the nature of the sketch structure, the estimated relative-frequency is never an underestimation, therefore all locations with at least θ relative-frequency are guaranteed to be in *Location-StreamSummary-Table*. $StreamSummary_{l_i}$ monitors the ϕ-frequent topics for location l_i. Since deletions need to be supported to maintain the list of ϕ-frequent topics for locations, this summary structure is also maintained thro-ugh a sketch-based solution. Consider a case where a pair (l_i, t_x) that expired is to be deleted from the data structures. StreamSummary$_{l_i}$ should only be updated to reduce $F(l_i, t_x)$, if (l_i, t_x) occurred after StreamSummary$_{l_i}$ was created. Therefore, StreamSummary$_{l_i}$ also includes a time-stamp TS_{l_i} recording the time it was created. In the case where the window size is set based on the maximum number of elements rather than real time, the timestamp will be based on the sequence number of pairs in S.

The second component given in Figure 1 is the *Topic-StreamSummary-Table*, a hashtable that monitors the topics that are potentially correlated with at least one location and a sketch structure to keep track of the topic frequencies. Through such an implementation, Premise 1 can also be addressed. The topics in this table are determined by the topics that appear in at least one $StreamSummary_{l_i}$ for location l_i that is θ-frequent in the current window. For each such topic t_x in *Topic-StreamSummary-Table*, there is a data-structure pair $< Count_{t_x}, StreamSummary_{t_x} >$ where $count_x$ is the number of locations t_x is ϕ-frequent for and $StreamSummary_{t_x}$ monitors the ψ-frequent locations for topic t_x. $StreamSummary_{t_x}$ will be maintained as long as $count_x$ is positive. As soon as this number reaches 0, the structure $StreamSummary_{t_x}$ is deleted freeing the space used by $< Count_{t_x}, StreamSummary_{t_x} >$. Similar to the stream summary structure for locations, $StreamSummary_{t_x}$ includes a time-stamp TS_{t_x} of when $StreamSummary_{t_x}$ was created.

An important sub-component of *GeoScope* that is leveraged in both *Location-StreamSummary-Table* and *Topic-StreamSummary-Table* is the sketch structure. This structure consists of a hashtable, *S[m][h]*, along with h hash functions. Given a range of elements from 1 to M, an item k in this range has a set of h associated counters and these counters are increased (or decreased) when encountering an insert (or delete) operation of element k. Clearly, the values for m and h should be set such that the collisions are minimized and guarantees can be given for bounds on overestimation. It has been shown that, $\frac{e}{\varepsilon} \cdot \ln(\frac{-M}{\ln p})$ counters are needed to estimate each item with error no more than εN in a window of size N with probability p by setting $m = \frac{e}{\varepsilon}$ and $h = \ln(\frac{-M}{\ln p})$ [16].

Given that the ϕ-frequent topics for a given location l_i are tracked only after l_i becomes θ-frequent and a topic t_x is tracked only after it becomes ϕ-frequent for at least one location, we need to show how *GeoScope* satisfies Premises 3, 1 and 2. To this end, we first give the intuition as to how these premises are still satisfied under our approximation. Premises 3 and 1 are relaxed to allow for a small error ε and to be guaranteed probabilistically. For this purpose, *GeoScope* requires two additional parameters ε and p in addition to the parameters θ, ϕ and ψ as described in Section 3.2. The parameter ε captures the allowed error rate while p captures the probability of remaining within this error rate.

In reference to Premise 1, instead of guaranteeing to capture the relative frequency of each topic and location exactly, *GeoScope* guarantees that for *any* topic t_x and any location l_i, its true relative frequency is overestimated by no more than ε with probability p but never underestimated. Note that theoretically, the ε and p values used to determine the error for locations and topics could potentially be distinct values. For ease of presentation we choose the same ε and p values for locations and topics. Also, in reference to Premise 3, even though an exact counter for each location is not kept, through the use of the sketch structure in *Location-StreamSummary-Table*, *GeoScope* guarantees detecting *all* locations l_i s.t. $F(l_i) \geq \theta N$. It also guarantees that no location l_j s.t. $F(l_j) < (\theta - \varepsilon)N$ is reported. Lastly, the relative frequencies of locations can be overestimated by no more than ε with probability p but never underestimated.

In reference to Premise 2, *GeoScope* guarantees capturing all *trending* correlated pairs of locations and topics rather than all correlated pairs. Here the notion of *trending* refers to a *non-decreasing* significance. Most importantly, *GeoScope* satisfies this premise deterministically which guarantees perfect recall values. While it is important to capture correlations in general, the really important task is to detect *trending* correlations, i.e. correlations that have an increasing value over time. For instance, consider two hypothetical correlations *(Los Angeles, 405 Traffic)* and *(Los Angeles, earthquake)*. Traffic in freeway 405 in Los Angeles is a general topic of interest resulting in a stable interest in the topic. In contrast, a recent hypothetical earthquake would result in *increasing* interest and therefore increasing value of correlation. While capturing both cases is important, it is crucial to *guarantee* capturing the latter. Even though *GeoScope* is only guaranteed to capture the trending correlations, as we will demonstrate in Section 5 it in fact captures all correlated pairs for various θ, ϕ and ψ settings. Similarly, even though there are no guarantees on the precision performance, as we show in Section 5, *GeoScope* provides near-perfect precision.

4.2 GeoScope Operations

Algorithm 1 Insert (l_i, t_x, ts)

1: $F(l_i) \leftarrow F(l_i)+1$
2: **if** l_i turned θ-frequent **then**
3: Create $StreamSummary_{l_i}$ with timestamp ts for location l_i
4: **if** l_i is θ-frequent **then**
5: $F_{l_i}(t_x) \leftarrow F_{l_i}(t_x)+1$
6: **if** t_x turned ϕ-frequent for l_i **then**
7: $StreamSummary_{l_i} = StreamSummary_{l_i} \cup \{t_x\}$
8: Increase $Count_{t_x}$
9: **for all** l_j turned θ-infrequent **do**
10: **for all** $t_y \in StreamSummary_{l_j}$ **do**
11: Decrease $Count_{t_y}$
12: Delete $StreamSummary_{l_j}$
13: **for all** t_y turned ϕ-infrequent for location l_i **do**
14: $StreamSummary_{l_i} = StreamSummary_{l_i} \setminus \{t_y\}$
15: Decrease $Count_{t_y}$
16: $F(t_x) \leftarrow F(t_x)+1$
17: **if** $t_x \in$ *Topic-StreamSummary-Table* **then**
18: $F_{t_x}(l_i) \leftarrow F_{t_x}(l_i)+1$
19: **if** l_i turned ψ-frequent for t_x **then**
20: $StreamSummary_{t_x} = StreamSummary_{t_x} \cup \{l_i\}$
21: **for all** l_j turned ψ-infrequent for t_x **do**
22: $StreamSummary_{t_x} = StreamSummary_{t_x} \setminus \{l_j\}$

There are three operations that are allowed at a given point: insert, remove and report. Each incoming stream element of the form (l_i, t_x) needs to be inserted into the data structure. As the sliding window moves along, expired mentions should be removed. Note that a sliding window can be set either in terms of number of elements to be maintained or the period of time defined in terms of minutes, hours, days etc. The pseudo-code for insert and remove operations is provided in Algorithms 1 and 2. Due to space limitations, we omit the pseudocode for the report algorithm that goes through the structures and reports correlated pairs.

In Algorithm 1, lines (1-15) perform updates due to the occurrence of l_i. Lines (1-8) capture the steps that need to be taken to incorporate the addition of the new mention in location l_i. In line 1, $F(l_i) \leftarrow F(l_i)+1$ entails updating the sketch structure that keeps track of location frequencies. Similarly, $F_{l_i}(t_x) \leftarrow F_{l_i}(t_x)+1$ (as given in line 5) entails updating the sketch structure in StreamSummary$_{l_i}$ to increase the value of t_x for location l_i. If t_x becomes ϕ-frequent for location l_i after this insertion, *Topic-StreamSummary-Table* needs to be updated to increase the number of locations t_x is trending for. If this count was zero before this operation, a new $StreamSummary_{t_x}$ will be created with timestamp ts and counter 1. Since the number of items increase with an insert operation, it is possible that a location whose frequency has not changed becomes θ-infrequent. Lines (9-12) remove such items and update the *Topic-StreamSummary-Table* for topics that were ϕ-frequent for such locations. Decreasing $Count_{t_y}$ also entails removing $StreamSummary_{t_x}$ if the counter becomes 0. Similarly, since the number of mentions in location l_i increases, there could be topics whose frequency has not changed and yet became ϕ-infrequent. Such cases are handled through lines (13-15). Starting from line 16, the changes to *Topic-StreamSummary-Table* are performed to capture the mention about topic t_x. First, the value of t_x is increased to satisfy Premise 1 as given in line 16. This entails updating the main sketch structure of *Topic-StreamSummary-Table*. Next, if t_x is already being tracked, $StreamSummary_{t_x}$ is updated to capture the new mention from location l_i.

In Algorithm 2 we present the steps that need to be taken for a remove operation. Here Lines (1-11) incorporate the reduction in the mentions from l_i while Lines (12-17) perform the deletion of t_x. Note that when an element is deleted the total number of elements in the given window decreases. In this case, there could potentially be a location l_j whose frequency is stable yet becomes θ-frequent. In order to avoid checking the frequency of each currently θ-infrequent location with every remove operation which would hurt the efficiency of *GeoScope*, we omit the creation of such $StreamSummary_{l_j}$. Even if such a summary were to be created, the set of topics in it would be empty. Therefore there is no penalty in omitting this action, the next time there is a mention from l_j, this stream summary will be created. The same is true for topics becoming ϕ-frequent for l_i, or locations becoming ψ-frequent for t_x. All such operations are omitted for efficiency purposes, while preserving precision guarantees.

Algorithm 2 Remove (l_i, t_x, ts)

1: $F(l_i) \leftarrow F(l_i)-1$
2: **if** l_i is θ-frequent **then**
3: **if** $TS(StreamSummary_{l_i}) \leq ts$ **then**
4: $F_{l_i}(t_x) \leftarrow F_{l_i}(t_x)-1$
5: **if** t_x turned ϕ-infrequent for l_i **then**
6: $StreamSummary_{l_i} = StreamSummary_{l_i} \setminus \{t_x\}$
7: Decrease $Count_{t_x}$
8: **if** l_i turned θ-infrequent **then**
9: **for all** $t_y \in StreamSummary_{l_i}$ **do**
10: Decrease $Count_{t_y}$
11: Delete $StreamSummary_{l_i}$
12: $F(t_x) \leftarrow F(t_x)-1$
13: **if** $t_x \in$ *Topic-StreamSummary-Table* **then**
14: **if** $TS(StreamSummary_{t_x}) \leq ts$ **then**
15: $F_{t_x}(l_i) \leftarrow F_{t_x}(l_i)-1$
16: **if** l_i turned ψ-infrequent for t_x **then**
17: $StreamSummary_{t_x} = StreamSummary_{t_x} \setminus l_i$

4.3 Running Time and Memory Requirements

Memory Requirements: A feasible geo-trend detection solution should be sub-linear in its space usage given the large scale of

data. In this section we provide proofs that *GeoScope* is sub-linear in both the number of locations and topics.

THEOREM 4.1. *GeoScope requires* $O(\frac{e}{\varepsilon*(\theta-\varepsilon)}(\ln(-\frac{|T|}{\ln(p)})+\frac{\ln(-\frac{|L|}{\ln(p)})}{\phi-\varepsilon})+\frac{1}{(\theta-\varepsilon)(\phi-\varepsilon)(\psi-\varepsilon)})$ *memory.*

PROOF. There are two main substructures: location table and topic table. The location table consists of the main sketch structure that tracks occurrences of locations in the window and requires $m_l * h_l$ counters. In order to fulfill Premise 3 that entails estimating the frequency of locations with error no more than εN with probability p, these values should be set as $m_l = \frac{e}{\varepsilon_l}$ and $h_l = \ln(-\frac{|L|}{\ln p_l})$ [16]. At a given time there are up to $\lceil\frac{1}{\theta-\varepsilon_l}\rceil$ locations being tracked for which a list of top topics should be maintained. For each of these $\lceil\frac{1}{\theta-\varepsilon_l}\rceil$ locations, $m_{lt} * h_{lt}$ counters are required for the sketch structure s.t. $m_{lt} = \frac{e}{\varepsilon_{lt}}$ and $h_{lt} = \ln(-\frac{|T|}{\ln p_{lt}})$ since pairs also need to be maintained to satisfy Premise 2. For each location, up to $\lceil\frac{1}{\phi-\varepsilon_{lt}}\rceil$ topics are tracked.

The second main substructure is for keeping track of important topics. The topics table consists of the main sketch structure that tracks occurrences of topics in a given window and requires $m_t * h_t$ counters. In order to fulfill Premise 1 that entails capturing topic frequencies correctly, these values should be set as $m_t = \frac{e}{\varepsilon_t}$ and $h_t = \ln(-\frac{|T|}{\ln p_t})$. For each tracked topic, a list of locations needs to be tracked. Since there are at most $\lceil\frac{1}{\theta-\varepsilon_l}\rceil$ locations tracked and for each location there are at most $\lceil\frac{1}{\phi-\varepsilon_{lt}}\rceil$ topics tracked, there are at most $\lceil\frac{1}{\theta-\varepsilon_l}\rceil\lceil\frac{1}{\phi-\varepsilon_{lt}}\rceil$ distinct topics in the topic table. For each of these topics, $m_{tl} * h_{tl}$ counters are required for the sketch structure s.t. $m_{tl} = \frac{e}{\varepsilon_{tl}}$ and $h_{tl} = \ln(-\frac{|L|}{\ln p_{tl}})$ since pairs also need to be maintained to satisfy Premise 2. In addition, there are at most $\lceil\frac{1}{\psi-\varepsilon_{tl}}\rceil$ locations tracked for each topic. Adding all these together, and simplifying the system by setting all $\varepsilon_{\{l,t,lt,tl\}} = \varepsilon$ and $p_{\{l,t,lt,tl\}} = p$, in total, the memory requirement sums up to $O(\frac{e}{\varepsilon*(\theta-\varepsilon)}(\ln(-\frac{|T|}{\ln(p)})+\frac{\ln(-\frac{|L|}{\ln(p)})}{\phi-\varepsilon})+\frac{1}{(\theta-\varepsilon)(\phi-\varepsilon)(\psi-\varepsilon)})$. □

Running time requirements: There are two possible update operations at a given time: an insert or a remove of a location-topic pair. Both of these operations have amortized log-linear running time. Due to space limitations, we skip the proof for the *remove* operation and note that it is very similar to the proof provided for the *insert* operation as provided below:

THEOREM 4.2. *The amortized running time for an insert operation in GeoScope is* $O(log(-\frac{|T|}{log(p)})+log(-\frac{|L|}{log(p)}))$

PROOF. The steps that need to be taken for an insert are given in Algorithm 1. Line 1 requires updating the sketch structure which entails $h = log(-\frac{|L|}{log(p)}))$ operations. Lines 2-3 create an empty stream structure if l_i becomes θ-frequent with the insertion of the new item. This clearly is a constant time operation. In the case where l_i was (or became) θ-frequent (Lines 4-8), $StreamSummary_{l_i}$ needs to be updated to include the addition of t_x. This entails updating $StreamSummary_{l_i}$ and possible insertions/deletions of the ϕ-frequent topics for l_i. Even with a conservative setting for the sketch structure that assumes all topics can be mentioned at a given location, the sketch update requires $h = log(-\frac{|T|}{log(p)})$ operations and the updates to the substructure is amortized-constant time. For the locations that have become θ-infrequent (Lines 9-12), the deletion operation is also constant time, however, with a non-constant number of such topics, the number of operations can become quite large. Since a location can only be deleted as many times as it is inserted to the stream summary and since by construction, a location l_j is inserted into the summary only when there is a tuple (l_j,t_y), we can conclude that the deletion operation has amortized constant time. Lines (13-15) requires amortized constant time for the same reason. In order to keep track of frequent global-level topics, sketch structure for topics is updated regardless of the topic being tracked or not requiring $h = log(-\frac{|T|}{log(p)})$ operations (Line 16). If topic t_x is being tracked (Lines 17-22), $StreamSummary_{t_x}$ has to be updated which entails updating the sketch structure for t_x (Line 18: $h = log(-\frac{|L|}{log(p)})$)), adding l_i to $StreamSummary_{t_x}$ if it became ψ-frequent (constant time) and deleting locations that became infrequent for t_x (amortized constant time). Adding all those operations together, amortized processing time for an insert is $O(log(-\frac{|T|}{log(p)})+log(-\frac{|L|}{log(p)}))$. □

4.4 GeoScope Accuracy Guarantees

Although *GeoScope* monitors the traffic of locations and topics approximately, its accuracy is very high. Here, we prove that *GeoScope* has *guaranteed* perfect recall in detecting *trending correlated pairs* where trending is defined based on non-decreasing relative frequency. The accuracy is defined w.r.t. the exact correlated pair computation. Before we delve into the proof of this statement, we introduce the notion of *trending* correlation.

DEFINITION 1. *A topic t_x is trending for location l_i if and only if $G_{[ts',ts]}(l_i,t_x) \le G_{[ts-w,ts]}(l_i,t_x)$ where $ts-w \le ts' \le ts$ and $G_{[ts_1,ts_2]}(l_i,t_x) = \frac{F_{[ts_1,ts_2]}(l_i,t_x)}{F_{[ts_1,ts_2]}(l_i)}$, $F_{[ts_1,ts_2]}(l_i,t_x)$ denotes the number of occurrences of the tuple between the time frames ts_1 and ts_2 and $F_{[ts_1,ts_2]}(l_i)$ denotes the number of occurrences of location l_i.*

DEFINITION 2. *A location l_i is trending for topic t_x if and only if $H_{[ts',ts]}(l_i,t_x) \le H_{[ts-w,ts]}(l_i,t_x)$, where $H_{[ts_1,ts_2]}(l_i,t_x) = \frac{F_{[ts_1,ts_2]}(l_i,t_x)}{F_{[ts_1,ts_2]}(t_x)}$, $F_{[ts_1,ts_2]}(l_i,t_x)$ denotes the number of occurrences of the tuple between the time frames ts_1 and ts_2 and $F_{[ts_1,ts_2]}(t_x)$ denotes the number of occurrences of topic t_x.*

Next, we define a *trending* correlated pair as follows:

DEFINITION 3. *A location-topic pair (l_i,t_x) is a trending correlated pair if and only if t_x is a trending topic for l_i and l_i is a trending location for t_x.*

Finally, we prove that *GeoScope* has perfect recall guarantee under the definition of *trending* correlated pairs. Although only *trending* correlations are *guaranteed* to be captured, we show in Section 5 that in practice *GeoScope* succeeds in detecting *all* correlated pairs. It also has a near perfect precision.

THEOREM 4.3. *At any given time ts, all trending correlated pairs in the time window ending at ts are reported by GeoScope.*

PROOF. Consider a particular time window that spans over the period $[ts-w,ts]$, where ts is the end of the window and w is the time window size and includes N tuples. Since (l_i,t_x) is a trending correlated pair, by definition $F(l_i) \ge \theta N$ and therefore l_i is guaranteed to be tracked. Let the time l_i starts being tracked be denoted by ts_{l_i} s.t. $0 \le ts_{l_i} \le ts$. Given the *trending* property, $G_{[ts_{l_i},ts]}(l_i,t_x) \ge G_{[ts-w,ts]}(l_i,t_x) \ge \phi * F_{[ts-w,ts]}(l_i)$. Therefore, topic t_x will also be tracked at a time $ts_{lt} \le ts$ which means t_x is guaranteed to be captured in the topics table. Since $H_{[ts_{lt},ts]}(l_i,t_x) \ge$

$H_{[ts-w,ts]}(l_i,t_x) \geq \psi * F_{[ts-w,ts]}(t_x)$, location l_i will also be tracked for topic t_x. Given the trending property, such frequencies will only increase in time guaranteeing that by ts, the pair will sustain its correlated property. □

5. EXPERIMENTS

Here, we first introduce the data set used in this study and provide a high level analysis. Next, we demonstrate the value of geo-correlated trends by focusing on the types of topics and locations that are detected by *GeoScope*. Finally, we evaluate the effect of parameters θ, ϕ, ψ as well as the window size on the accuracy and efficiency of *GeoScope* . Throughout the experiments we chose $\varepsilon = 0.0004$ and $p = 0.99$ to allow for small error.

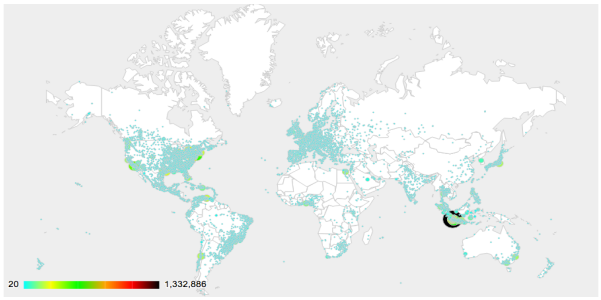
5.1 GeoScope Case Study: Twitter

Given the widespread use of Twitter, we demonstrate the usefulness of *GeoScope* on the Twitter platform. Since tweets are restricted to at most 140 characters long, Twitter users use hashtags to convey their thoughts in a compact manner [29]. Therefore, we choose *hashtags* to capture topics. As for the definition of locations, we focus on *cities* since this resolution is large enough to capture local interests and not too small to result into meaningless correlations. It also results in interpretable results that map to real events happening in different cities of the world. We note that *GeoScope* can easily be adapted to other topic definitions such as n-grams, urls and named entities. Similarly, locations can be defined as states or arbitrary bounding boxes. We skip such analysis due to space limitations and leave such a broad exploration as future work.

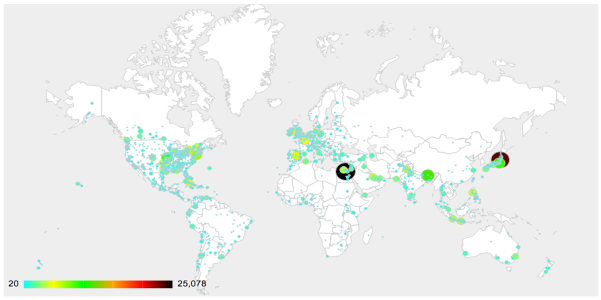
For our experiments we used Twitter statuses (tweets) from February 1st to June 18th 2011. The data is extracted through Twitter's public API (GardenHose) and constitutes $\sim 10\%$ of the overall Twitter statuses of that time period. The average number of tweets per day is 14.2M (with a total of 2 billion for the whole period). After geo-tagging, a procedure described below, we obtained a total of 378,941,219 labeled datapoints, out of which 63M also include a hashtag. The number of unique users in our data set is 46M. The geographical data was obtained from [23], which contains complete hierarchical information and coordinates for approximately 50,000 cities from all the countries and regions of the globe.

Geo-tagging Twitter Content: There are two types of geographical information that can be associated with a given tweet: the location the tweet is shared from (*geo-origin*) and the location the tweet is about (*geo-focus*). Our trend detection solution can be applied to both cases. In this paper, we focus on geo-trends when location is based on *geo-origin* and skip the analysis of *geo-focus* due to space limitations. To identify the *geo-origin* of a tweet, we utilize two signals: tweet and user location. Tweet location is provided explicitly by the Twitter API in the form of a latitude and longitude pair. However, only 1.5% of tweets' *geo-origin* are identified through this method. The second signal, user location, is a user provided free-form text that carries more noise [36]. We extract this information by parsing the location string and identifying pairs of (longitude, latitude), (city-name, region-abbreviation), (city-name, region-name), (city-name, country-abbreviation) and (city-name, country-name). In cases of city name ambiguity, we choose as the best match the one with the largest population. After obtaining the location of a user, all her untagged tweets are tagged with this location, which increases the percentage of tagged tweets according to their *geo-origin* to 13%.

Geographical Distribution of Twitter Updates: We provide heat maps of locations that tweets originate from (Figure 2(a)) and locations tweets are about (Figure 2(b)) to provide an overview of



(a) Tweets *in* Cities



(b) Tweets *about* Cities

Figure 2: Heat Map for # of tweets in/about cities of the world

the geographical characteristics of our data set. In both maps, we plot every city associated with more than 10 tweets using the GeoMap tool of Google Charts. The color and size of cities are proportional to the number of tweets. The two figures resemble each other but there are certain distinctions. For instance, Japan is denser in Figure 2(b) due to the Japan earthquakes that took place within the time period captured in our data set. On the contrary, a drop in significance can be observed for countries such as Indonesia when comparing the tweets *in* cities to tweets *about* cities. This difference is due to the fact that Indonesia is a highly active country in Twitter [15], while there are no events with global implications taking place in its cities. Note that we also analyzed the number of users per location and the results were similar to Figure 2(a) and are omitted due to space limitations.

5.2 Effectiveness of GeoScope

We evaluate the effectiveness of *GeoScope* through various human validation tasks. We compare our solution to three baselines:

Traditional Heavy-Hitters Approach (*THHA*): Detects the most popular topics in the entire stream.

Geographical Heavy-Hitters Approach (*GHHA*): Simply reports the ϕ-frequent topics for all θ-frequent locations. Unlike *GeoScope*, *GHHA* does not filter pairs where the location is not significant for the topic (no ψ parameter).

Statistically Significant Topic-Location Detection (*SSTLD*): Defines the association strength between a location and a topic by comparing their *expected* and *observed* frequencies. For this purpose, we compute the χ^2 statistic for each topic-location pair.

Given these three baseline approaches, our goal is to evaluate the geographical significance of topics they detected. We focus on two high level characteristics:

1. Information Overload: This notion refers to a state in which the overwhelming amount of information leads to communication inputs not being processed and utilized [28]. Therefore, it is important not to overwhelm users by reporting too many topic-location

	THHA vs. *GeoScope*	*GHHA* vs. *GeoScope*	*SSTLD* vs. *GeoScope*
fraction of *GeoScope* hashtags	0.94	0.74	0.89
Fleiss Kappa	0.64	0.51	0.65
# of judges	10	10	9

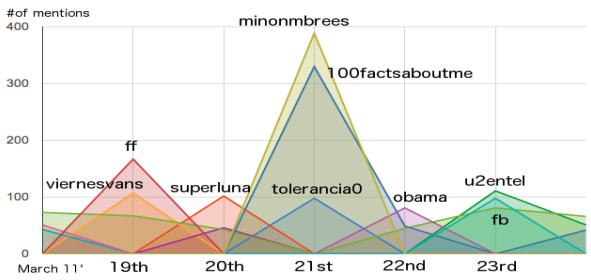
Table 1: Human Validation Results

pairs. Our experiments reveal that there are approximately 17, 23, and 150000 pairs that would be reported by *GeoScope*, *GHHA* and *SSTLD* respectively when the parameters are set as $\theta = 0.005, \phi = \psi = 0.05$ and the sliding window is set to 24 hours. Clearly, *GeoScope* provides the most manageable list with the smallest number of pairs to inspect, reducing the effects of information overload.

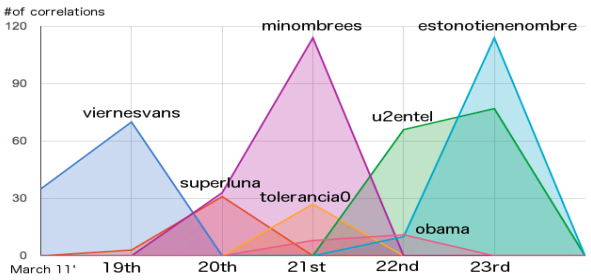
2. Information Quality: Having established the superiority of *GeoScope* along the information overload dimension, we now focus on information quality. Our goal is to identify the likelihood of a user preferring a topic reported by *GeoScope* to one that would be reported by each of the baseline methods w.r.t. geographical significance. For this purpose, we use human judges. Each judge is presented an online questionnaire. Each question presents the judge with two hashtags h_1 and h_2 and asks her to identify the one that carries the most geographical significance. The judge is not provided any context as to how the hashtags are chosen. We created 3 questionnaires to compare: {*THHA* vs. *GeoScope*}, {*GHHA* vs. *GeoScope*}, and {*SSTLD* vs. *GeoScope*}. The set of h_1, h_2 hashtag pairs to include in the questionnaire is selected as follows: We consider the most commonly reported topics (hashtags) for each baseline and *GeoScope* and retrieve the top 100 topics from each list. We first eliminate the common topics between the two methods being compared and randomize the order of topics reported by each method. Next, we randomize the order in which h_1 and h_2 are presented such that with 0.5 probability the hashtag identified by *GeoScope* is presented as the first (or second) option. Through such randomization, we make sure that the only information the judges process in making a decision is inherent to the topic itself.

We launched the 3 questionnaires online and requested participation from computer science graduate students and faculty. Our findings are represented in Table 1. The three columns correspond to the three tasks. The first row reports the fraction of questions for which the majority vote selected the *GeoScope* hashtag as the hashtag with more geographical significance. For two indistinguishable methodologies this fraction should lie around 0.5. We can see that the judges prefer *GeoScope* to all of the 3 baselines by a large margin. The closest approach is *GHHA*. Even in this case, the judges prefer *GeoScope* to *GHHA* approximately 4:1. We also computed the inter rater reliability measures for each of the tasks using Fleiss Kappa (FK) [10]; a statistical measure commonly used for assessing the reliability of agreement between a fixed number of judges when assigning categorical ratings. We report our findings on the inter rater reliability measure on the second row of Table 1. The Fleiss Kappa measure ranges from 0 to 1 and higher values indicate better agreement. Kappa measure of ≥ 0.6 and $0.4 - 0.6$ are considered to capture *substantial* and *moderate* agreement respectively. Therefore we conclude that the distinction between methodologies, or the topics detected by them, is clear.

Figure 3 provides more insights into the distinction between *GeoScope* and *GHHA*. Figure 3(a) displays the *GHHA* results in the city of Santiago, Chile for a period of 5 days (March 19-23 2011). The y-axis represents the popularity of a particular hashtag that is mentioned. We restrict this timeline to 5 days and the top-3 topics due to



(a) *GHHA*



(b) *GeoScope*

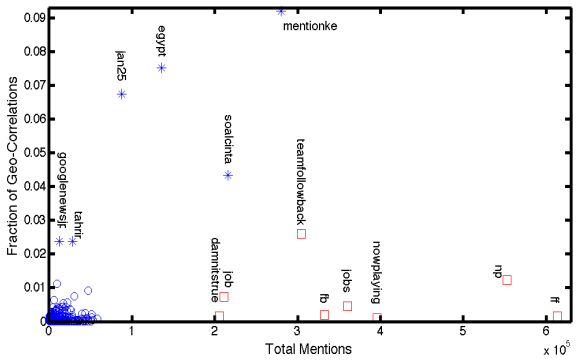
Figure 3: Trends detected in Santiago using two methods

the large number of topics detected using the simple *GHHA*. Figure 3(b) provides the list of topics which are detected by *GeoScope* as correlated with Santiago for the setting of $\theta = 0.005, \phi = 0.05$ and $\psi = 0.05$. Here the y-axis is the number of times the topic is detected as correlated in this particular location (reporting frequency of 15 minutes). The difference between the two Figures demonstrates a number of interesting characteristics. For one, *GHHA* reports an overwhelmingly large number of topics even when reporting up-to 3 topics in each day. Second, topics with no geo-intend like *#ff, #fb, #100factsaboutme* are not reported by *GeoScope*. We omit similar graphs which compare *GeoScope* to *THHA* and *SSTLD* since these two techniques showed poor performance as demonstrated in Table 1.

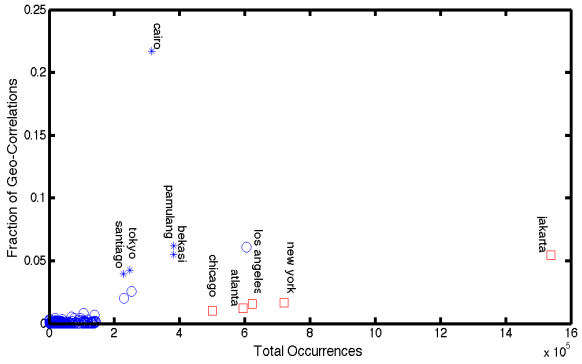
5.3 GeoScope topics and locations

Here, we address the following two questions through analysis enabled by *GeoScope*: *Are there topics that carry a higher geo-significance?* and *Are there locations that exhibit local topical interests?* To address the first question, in Figure 4(a) we show the relation between the geo-significance of topics and the total number of times they are mentioned in the data set measuring their global importance. The geo-significance of a topic is measured in terms of the fraction of all the correlated pairs it appears in the entire stream. We chose a time window of 24 hours in this experiment and set $\theta = 0.005, \phi = \psi = 0.05$. *GeoScope* provides means for reporting correlated pairs at any given time. For this experiment we chose 10 minutes as the reporting frequency. Note that the reporting frequency and the time window are two distinct values. The time window refers to the length of the sliding window while the reporting frequency reflects how frequently the report operation is called to determine the current list of correlations.

For ease of viewing, we eliminated all hashtags that had no correlations reported, which reduced the number of data points drastically from over 2 million to approximately 250. This indicates that even though there is a large number of topics discussed in social networks, there is only a small number of topics that carry signif-



(a) Hashtags



(b) Cities

Figure 4: Geo-significance vs. trendiness of hashtags and cities

icance in different localities. There are various hashtags that have high global significance while being much less important as geographical trends such as #ff, #np, #jobs (represented by squares in Figure 4(a)). For instance, #ff refers to "follow friday" and is a popular hashtag used in Twitter. Similarly, #jobs, referring to issues related to jobs, is a common hashtag that is of interest to Twitter users in the global scale. Unlike these topics that are of interest to the entire network, hashtags such as #jan25, #egypt, #googlenewsjp (represented by stars in Figure 4(a)) are a lot more significant as a geographical trend. The first two of these hashtags relate to recent uprisings in Egypt while #googlenewsjp is mostly used to discuss issues about the Fukushima earthquake in April 2011.

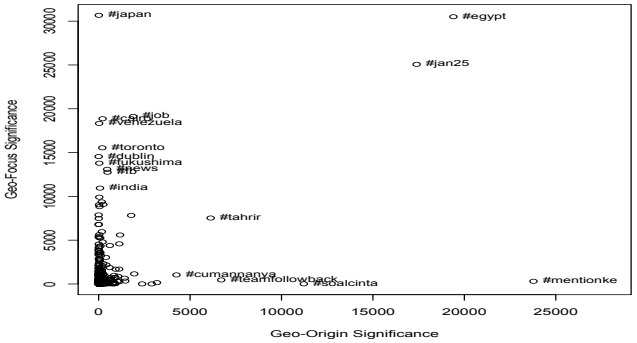
We performed a similar analysis to capture which cities carry geo significance, i.e. cities whose residents are interested in local topics. For this purpose we plot the number of correlations a given city appears in versus the number of tweets originating from that particular city. As can be seen from Figure 4(b), the static representation of a city measured by the number of tweets originating from it, is not representative of the geo-significance of that place. For instance, there is a relatively small number of tweets originating from Cairo but due to those tweets being mostly about local events (the recent political uprising) they have high geo-significance. Another city with a large number of geo-correlations is Santiago. Examples of detected correlations for this city include sports related hashtags (e.g. #bielsa) and cultural events and TV programs (e.g. #wewantsupershowinlatinoamerica). On the contrary, we see that Jakarta, a city where a large number of identified users reside, does not appear in a large number of correlations, meaning that users from this area are in general less concerned about local events.

The analysis provided so far focused on the cumulative geo-significance of topics and locations, but *GeoScope* provides a more useful tool that can capture geo-significance of topics or locations along a temporal dimension as well, by detecting correlations along a sliding window. There is a large number of interesting topics detected at particular points in time but do not appear in Figures 4(a) or 4(b) due to their short lived activity. A few examples include: Iwaki aftershock on April 11, as well as the main Japan earthquake on March 11. On these days the hashtag #earthquake is detected to be correlated with Tokyo due to a large number of Twitter users from Tokyo mentioning this topic. Such behavior indicates that *GeoScope* can be used in crisis management as it detects emergency events in a fast and automated manner. However, local interests detected by *GeoScope* are not only limited to emergency events. Not only can the political interests of a population be captured as in the case of correlated pairs, such as *(Cairo, #Jan25)*, but it can also capture other, more casual interests. For instance, a large number of correlated pairs involving Soccer teams appear in British cities, especially compared to other cities of the world, indicating a high British interest in this sport. Examples of this type of correlations include *(Nottingham, #NewCastle)* or *(Liverpool, #lfc)*.

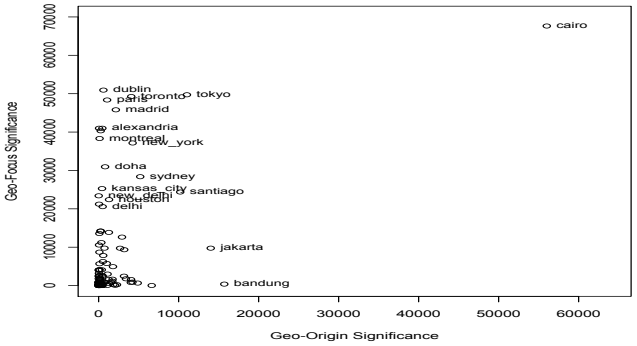
Local and short lived events, such as political demonstrations and cultural events, are also among the topics captured by *GeoScope*. As an example, the correlated pair *(Madrid, #11m)* is captured due to the demonstrations in Madrid on the anniversary of bombings that happened on March 11 2004, killing 191 people. Examples of detected cultural events include the correlated pair *(Austin, #sxsw)* that is due to the SXSW festival on March 16 2011 in Austin. Other correlation pairs appear in the general form of *(city, #city)*. This is due to the fact that Twitter users use hashtags to geo-tag and organize important information, especially in the case of emergencies [32]. Note that the correlations detected are currently restricted by the use of hashtags as topics. As future research direction, we aim to investigate determining significant keywords in tweets and using them as topics as well.

Geo-Origin vs. Geo-Focus: As stated before, *GeoScope* provides a flexible framework, allowing for various definitions of locations (cities, states or arbitrary bounding boxes) as well as topics (hashtags, named entities or n-grams). In addition, locations can refer to where the social content is created (*geo-origin*) or what location the social content is about (*geo-focus*). While space limitations prevent us from presenting findings for the entire spectrum of such settings, we provide some high level findings from our analysis of the application of *GeoScope* where the location is defined based on *geo-focus*. Identifying the *geo-focus* of a tweet requires detecting locations mentioned in the tweet content in a fashion similar to detecting user locations from the location field. The percentage of tweets whose *geo-focus* we identified through this method is 8%.

Figure 5 presents the distinction between the *geo-origin* and *geo-focus* analysis. The analysis reveals various interesting distinctions. For instance, as seen in Figure 5(b), cities such as Jakarta and Bandung that commonly talk about local topics (high *geo-origin* significance) have low *geo-focus* values indicating low international recognition. We see a boost in significance for cities such as Tokyo which can be attributed to the international interest in the Japan earthquake. Similar characteristics can be observed from Fig 5(a). Hashtags on globally important yet local events (e.g. #japan) and hashtags that report on local events (#jobs for reporting job openings in various cities) see a boost in significance in *geo-focus*. Topics that attract both significant attention from the location they originate as well as the rest of the world (#jan25) have similar *geo-focus* and *geo-origin* significance.



(a) Hashtags



(b) Cities

Figure 5: Geo-Origin vs. Geo-Focus importance of hashtags and cities

5.4 The Accuracy of GeoScope

Here, we investigate the accuracy of *GeoScope* w.r.t. the exact method introduced in §3.3. We first start by examining the number of correlated pairs detected with varying values of ϕ and ψ. As can be seen in Figure 6, increasing ϕ and ψ drastically decreases the number of correlated pairs. To evaluate the effect of changing ϕ, the other two parameters were set as $\theta = 0.005$ and $\psi = 0.05$, while varying ϕ between 0.005-to-1. Similarly, evaluating the effect of changing ψ, the other parameters were set as $\theta = 0.005$ and $\phi = 0.05$ while varying ψ between 0.005-to-1. The difference is more significant for small ϕ values, which indicates that it is less likely for the entire population to be interested in *only* one topic, while it is far more likely that there is *only* one (or few) location(s) that is interested in a given topic. Note that this artifact is somewhat created by design; the limitation on θ filters extremely inactive locations with few users whose interests can be extremely focused. These experiments provide a guide to the ***right choice of dominance(ϕ), support(ψ) and θ values*** since one can make parameter choices based on the number of correlations that they aim to capture at a given time. However, we would like to point out that the proper settings for these values are dependent on the social network studied as well as the specific application. Therefore, our goal is to provide a general framework that can meet different needs rather than defining one set of parameter settings that is globally optimal.

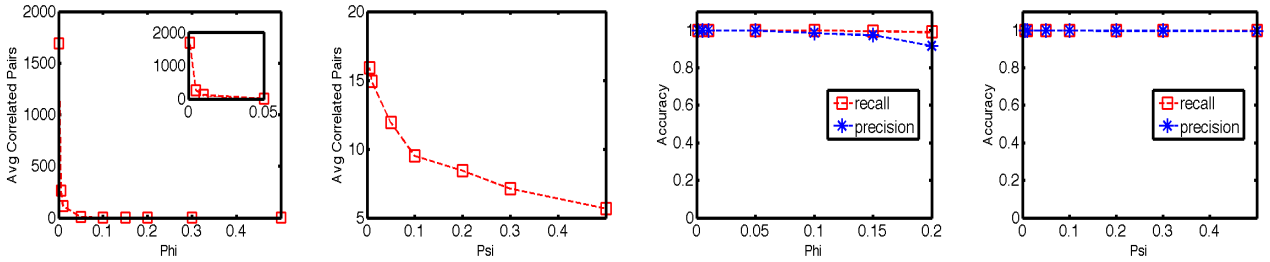
Next, we examine how varying ϕ and ψ affects the recall and precision of *GeoScope*. As stated in Theorem 4.3, *GeoScope* is guaranteed to capture all the *trending* correlated location-topic pairs, where trendiness is defined based on a non-decreasing frequency function. We now show two important findings: first, *GeoScope* succeeds in capturing correlated location-topic pairs that do not necessarily follow this strict distribution and second, in addition to recall, *GeoScope* 's precision is very high. As shown in Figures 6(c) and 6(d), *GeoScope* has a perfect recall rate over various settings for ϕ and ψ values while the precision rate is slightly affected by increasing ϕ. The results in Figure 6(c) are obtained by setting the time window to 24 hours, $\theta = 0.005$, $\psi = 0.05$ and varying ϕ. Similarly, the results provided in Figure 6(d) are obtained by setting the time window to 24 hours, $\theta = 0.005$ and $\phi = 0.05$ and varying ψ. Due to space limitations we omit the figures showing the behavior of *GeoScope* with varying θ values. The analysis shows that the number of correlated pairs drops drastically with increasing θ.

5.5 Space and Time Efficiency of GeoScope

Space Efficiency of *GeoScope*: In Figure 7(a), we provide a comparison between the exact solution and *GeoScope*. The space comparison is based on the number of counters used by the two methods. For the exact solution this value would be equivalent to the number of unique elements while for *GeoScope* it captures the number of elements maintained in the trending lists as well as the memory used for the sketches. Results provided in Figure 7(a) are based on the settings $\theta = 0.05$, $\phi = \psi = 0.1$ but we note that the general trend is similar for various other settings as well. *GeoScope* provides means for defining the window size in terms of actual time or the number of elements to be maintained. For the purpose of this experiment, as our goal is to capture how well the algorithms scale, the window size is defined based on the number of elements. The recent numbers published by Twitter claim an average of 400 million tweets per day [34]. Therefore a geo-trend detection mechanism that aims to capture daily trends should process 400 million elements on average. We performed experiments setting the window size to 1, 2.5, 5, 7.5, 10, 15 and 20 million respectively. The results can be seen in the *inner subfigure* in Figure 7(a). Here the X-axis represents the window size while the y-axis represents the number of counters needed to execute each method. This figure shows that the memory requirement of the exact method increases with window size while it remains constant for *GeoScope*.

Given the outcome of these experiments we extrapolate the memory requirements when this number reaches 400 million. Note that we tried both linear and logarithmic fit for the increase in memory requirements and determined the best fit according to the R^2 measure. R^2 statistic takes on values between 0 (poor fit) and 1 (good fit) and indicates how closely values obtained from fitting a model, match the dependent variable the model is intended to predict. The R^2 measure for the memory requirements of linear regression is 0.9956, while this number is 0.8870 for logarithmic fit, showing that a linear increase best describes the trend. Given this result, we demonstrate the memory needed to process 400 Million tweets in a given time window in Figure 7(a) by a vertical dashed line. Memory usage of the exact solution is larger than *GeoScope* even for small window sizes. However, the difference is more pronounced as the window size gets larger since the memory requirement of the exact solution increases while *GeoScope* is unaffected.

Time Efficiency of *GeoScope*: In satisfying Premises 3, 1 and 2, *GeoScope* answers three types of queries at any particular time: reporting on frequencies of locations (Premise 3), frequency of topics (Premise 1) and reporting on correlated pairs (Premise 2). The efficiency of *GeoScope* in answering queries relating to Premises 3 and 1 can be directly inferred from the results of heavy-hitters approaches and more specifically the sketch based method we use



(a) Effect of ϕ on # pairs (b) Effect of ψ on # pairs (c) Effect of ϕ on accuracy (d) Effect of ψ on accuracy

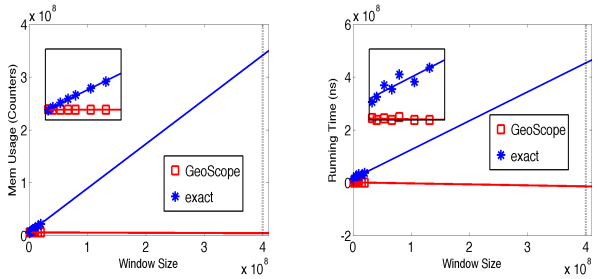
Figure 6: Effect of ϕ and ψ in the average number of correlated pairs detected by *GeoScope* and accuracy measures

as a building block [16]. Due to space limitations, we omit such analysis and focus on the efficiency of *GeoScope* in reporting correlated pairs. The three types of operations of interest are; *insert*, *remove* and *report*. In Figure 7(b), we present a similar analysis to the one presented for the space usage with identical settings for the parameters ($\theta = 0.05, \phi = \psi = 0.1$), while we note that the results are similar for other settings. We present both the results of running timing experiments for window size to 1, 2.5, 5, 7.5, 10, 15 and 20 million (the inner subfigure in Figure 7(b)) as well as the timing requirements for when the window size is extrapolated to 400 million (the main figure in 7(b)). Note that we used *getCurrentThreadUserTime* method in *ThreadMXBean* class in Java for recording time, which does introduce a certain level of noise.

As the number of elements in the time window increases, the time required to report on the correlated pairs increases linearly for the exact solution while *GeoScope* is not affected. Similar to Figure 7(a), we mark the 400 million point that corresponds to the average number of tweets per day. The results show that the exact solution does not scale. Also note that this linear fit is under the assumption of limitless memory. In reality as the number of elements increase in the given window, the memory required for the exact solution increases drastically. Implementing the exact solution in a real system with memory limits would result in thrashing which in turn increases run time drastically. Similar analysis was performed to test the efficiency of the *insert* and *remove* methods. Unlike with the report method, these methods scale nicely with increasing window size for both exact solution and *GeoScope*. As the window size increases, update methods involve updating already existing structures more often than creating and destroying counters, resulting in such a performance. In general, our experiments show that the update performance of *GeoScope* is comparable to the exact solution, but for certain parameter settings the exact solution slightly outperforms *GeoScope*. Note again, that this analysis is performed assuming unlimited memory. By decreasing the memory available, update methods of the exact solution would also result in thrashing and consequently worse running time, while such limitation does not affect *GeoScope*.

6. CONCLUSION

Geography plays an important role in our lives, shaping the friendships we form, and the interests we develop. The significance of geography in data analysis is clear since "...near things are more related than distant things" as the first law of geography states. Such significance incidentally also exists in the virtual extension of our daily lives; online social networks where users tend to befriend people and talk about events that are close-by. However, studying social networks through geo glasses goes well beyond a simple intellectual exercise. Recent events have shown that online



(a) Memory Usage (b) Report Running Time

Figure 7: Memory usage and Running Time comparison

social networks can be used in the case of a crisis to first *detect* the emergency event and later to deliver important information to interested users. Due to the large amount of noisy data shared on social networks, the *detection* of such significant *local* events becomes a non-trivial problem. Therefore, it is critical to provide large-scale data analysis tools that analyze social networks from a geographical perspective and detect such local events or interests in an online manner by also capturing the temporal aspects of information trends. This undertaking is the main focus of our study.

To this end, in this work we studied the online detection of geo-correlated information trends, i.e. identifying correlated location-topic pairs along a sliding window in a social data stream. We showed that the exact solution for such a problem requires keeping track of all possible pairs of location-topic pairs which is infeasible due to the large scale of data. Therefore, we introduce *GeoScope*: an approximate solution that requires only sub-linear memory and running time while guaranteeing to capture all *trending* correlations. We experimentally studied the value, accuracy and efficiency of *GeoScope* in Twitter and showed that this tool provides a manageable list of interesting location-topic pairs including crisis events such as earthquakes, or local events such as political demonstrations, concerts or sports events. The experiments show that *GeoScope* scales well with increasing amount of data while the exact solution suffers from such an increase. Moreover, the experiments show that, in addition to perfect recall measures, *GeoScope* also has a high precision.

Even though in our experiments we apply *GeoScope* to detect trends in Twitter, the tool is generic enough to be used in other social networks as well. Similarly, the topics, as defined based on hashtags in this study, or locations, defined based on cities, can be redefined. In fact, topic detection of information items shared in social networks is an important open problem which can reshape how a topic is to be defined in *GeoScope*. Similarly, locations of

interests can be regions, countries or simply arbitrary polygons on a map. *GeoScope* can easily be used to detect geo-trends in all these resolutions. An important future work in this context is to detect hierarchical geo-trends by capturing the right resolution in which a topic is trending in an online manner. Although multiple *GeoScope* structures can be used in parallel to address this problem, our future goal is to investigate if there are more compact ways in which hierarchical geo-trend detection can be performed.

7. ACKNOWLEDGEMENTS

This work is partially supported by NSF Grant IIS- 1135389 and a gift from the Bill and Melinda Gates Foundation.

8. REFERENCES

[1] N. Bansal and N. Koudas. Blogscope: a system for online analysis of high volume text streams. In *VLDB '07*, pages 1410–1413. VLDB Endowment, 2007.

[2] C. Budak, D. Agrawal, and A. El Abbadi. Structural trend analysis for online social networks. *Proc. VLDB Endow.*, 4:646–656, July 2011.

[3] H. Cao, G. Hripcsak, and M. Markatou. A statistical methodology for analyzing co-occurrence data from a large sample. *J. of Biomedical Informatics*, 40(3):343–352, 2007.

[4] M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent elements in data streams. In *ICALP'02*, pages 693–703, 2002.

[5] Z. Cheng, J. Caverlee, and K. Lee. You are where you tweet: a content-based approach to geo-locating twitter users. In *CIKM '10*, pages 759–768. ACM, 2010.

[6] G. Cormode and S. Muthukrishnan. What's Hot and What's Not: Tracking Most Frequent Items Dynamically. *TODS'05*, 30(1):249–278, 2005.

[7] A. Dalli. System for spatio-temporal analysis of online news and blogs. In *WWW '06*, pages 929–930, New York, NY, USA, 2006. ACM.

[8] E. D. Demaine, A. López-Ortiz, and J. I. Munro. Frequency estimation of internet packet streams with limited space. In *ESA'02*, volume 2461, pages 348–360, 2002.

[9] J. Eisenstein, B. O'Connor, N. A. Smith, and E. P. Xing. A latent variable model for geographic lexical variation. In *EMNLP '10*, pages 1277–1287, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[10] J. L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76:378–382, 1971.

[11] M. Gardner and D. Altman. *Statistics with confidence: confidence intervals and statistical guidelines*. Brithis Medical Journal, 1995.

[12] N. S. Glance, M. Hurst, and T. Tomokiyo. Blogpulse: Automated trend discovery for weblogs. In *WWE 2004*. ACM, 2004.

[13] S. Havre, B. Hetzler, and L. Nowell. ThemeRiver: visualizing theme changes over time. In *InfoVis 2000*, pages 115–123, 2000.

[14] L. Hong, A. Ahmed, S. Gurumurthy, A. J. Smola, and K. Tsioutsiouliklis. Discovering geographical topics in the twitter stream. In *WWW '12*, pages 769–778, 2012.

[15] Indonesia, brazil and venezuela lead global surge in twitter usage. `http://www.comscore.com/Press_Events/Press_Releases/2010/8/Indonesia_Brazil_and_Venezuela_Lead_Global_Surge_in_Twitter_Usage`.

[16] C. Jin, W. Qian, C. Sha, J. X. Yu, and A. Zhou. Dynamically maintaining frequent items over a data stream. In *CIKM '03*, pages 287–294. ACM, 2003.

[17] H. Kwak, C. Lee, H. Park, and S. Moon. What is Twitter, a social network or a news media? In *WWW '10*, pages 591–600, 2010.

[18] T. Lappas, M. R. Vieira, D. Gunopulos, and V. J. Tsotras. On the spatiotemporal burstiness of terms. *Proc. VLDB Endow.*, 5(9):836–847, May 2012.

[19] J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *KDD '09*, pages 497–506, 2009.

[20] A. M. MacEachren, A. C. Robinson, A. Jaiswal, S. Pezanov, A. Savelyev, J. Blanford, and P. Mitra. Geo-Twitter analytics: Application in crisis management. In *25th International Cartographic Conference*, July 2011.

[21] G. S. Manku and R. Motwani. Approximate frequency counts over data streams. In *VLDB'02*, pages 346–357, 2002.

[22] M. Mathioudakis and N. Koudas. Twittermonitor: trend detection over the twitter stream. In *SIGMOD '10*, pages 1155–1158, New York, NY, USA, 2010. ACM.

[23] Maxmind world cities with population. `http://www.maxmind.com/app/worldcities`.

[24] A. Metwally, D. Agrawal, and A. El Abbadi. An integrated efficient solution for computing frequent and top-k elements in data streams. *TODS'06*, 31(3):1095–1133, 2006.

[25] A. Metwally, F. Emekçi, D. Agrawal, and A. El Abbadi. Sleuth: Single-publisher attack detection using correlation hunting. *Proc. VLDB Endow.*, 1(2):1217–1228, Aug. 2008.

[26] B. Poblete, R. Garcia, M. Mendoza, and A. Jaimes. Do all birds tweet the same?: characterizing twitter around the world. In *CIKM '11*, pages 1025–1030. ACM, 2011.

[27] W. M. Pottenger and T.-h. Yang. Detecting emerging concepts in textual data mining. *Computational information retrieval*, 100, 2001.

[28] E. M. Rogers and R. Agarwala-Rogers. Organizational communication. *Communication behaviour*, pages 218–239, 1975.

[29] 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 *WWW '11*, pages 695–704. ACM, 2011.

[30] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *WWW '10*, pages 851–860. ACM, 2010.

[31] J. Sankaranarayanan, H. Samet, B. E. Teitler, M. D. Lieberman, and J. Sperling. Twitterstand: news in tweets. In *GIS '09*, pages 42–51, 2009.

[32] K. Starbird and L. Palen. "voluntweeters": self-organizing by digital volunteers in times of crisis. In *CHI*, pages 1071–1080, 2011.

[33] W. Tobler. A computer movie simulating urban growth in the detroit region. *Economic Geography*, 46(2):234–240, 1970.

[34] Twitter hits 400 million tweets per day. `http://news.cnet.com/8301-1023_3-57448388-93`.

[35] J. Ugander, B. Karrer, L. Backstrom, and C. Marlow. The anatomy of the facebook social graph. *CoRR*, 2011.

[36] B. Wing and J. Baldridge. Simple supervised document geolocation with geodesic grids. In *ACL*, 2011.

[37] Y. Zhu and D. Shasha. Statstream: statistical monitoring of thousands of data streams in real time. In *VLDB '02*, pages 358–369, 2002.