Event Detection in High Throughput Social Media

Michael Weiler

München 2016

Event Detection in High Throughput Social Media

Michael Weiler

Dissertation
an der
Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität
München

vorgelegt von
Michael Weiler
aus München

München, den 20.10.2016

Erstgutachter: Prof. Dr. Hans-Peter Kriegel
Zweitgutachter: Prof. Dr. Michael Gertz
Mündliche Prüfung: 20. Dezember 2016

Eidesstattliche Versicherung

(Siehe Promotionsordnung vom 12.07.11, § 8, Abs. 2 Pkt. .5.)

Hiermit erkläre ich an Eidesstatt, dass die Dissertation von mir selbstständig, ohne unerlaubte Beihilfe angefertigt ist.

Michael Weiler

Name, Vorname

....

Ort, Datum   Unterschrift Doktorand/in

Contents

List of Figures

List of Tables

Abstract

Social media platforms like Twitter or Facebook are a popular source for live textual data. Together with daily articles from newspapers and web blogs they yield an overwhelming amount of data. This flood of information makes it infeasible for individuals to keep the overview beyond a certain level of popularity (like the global top stories of popular news papers).

By detecting emerging topics, data could be summarised by its corresponding series of events, which makes it easier for users to understand what actually happens regarding multiple topics at the same time. But analysing such data is challenging: Algorithms need to handle very large, rapidly changing and fast arriving data sets that are heterogeneous, consisting of text, geographic locations and images. Due to the growing need for summarisation and abstraction combined with the increasing availability of public textual data sources, the field of emerging topics detection has recently gained a lot of attraction in research as well as in industry. However, existing methods are often limited in what they track: They require a user to specify a set of keywords in advance, operate only on a limited subset like hashtags, or are only able to detect events of global magnitudes such as the NFL Super Bowl or natural disasters.

In this thesis, we will provide contributions in the field of emerging topics detection with respect to text as well as geo-spatial data. As a first contribution we introduce a method capable of processing vast amounts of data using an efficient online algorithm, which is nevertheless able to identify unusual textual patterns in the data stream without requiring the user to specify any constraints in advance (such as hashtags). We therefore propose a significance measure that can be used to detect emerging topics early, long before they become "hot tags", by drawing upon experience from Outlier Detection. By using hash tables in a heavy-hitters type algorithm for establishing a noise baseline, we show how to track even all word pairs using only a fixed amount of memory. By comparing the frequencies of newly observed values to statistics from earlier data, we can immediately report significant deviations with minimal delay, which makes our system capable of reporting "Breaking News" in real-time as they happen in social media.

As a second contribution, we address the unequally distributed adoption in social media, which leads to distorted events when not considered carefully (e.g. in our 1 year Twitter sample, London holds a 0.67% share that is more than twice as large as *all* of Germany's 0.31% share). By using geo-enriched

textual data we estimate each geographic region's local density, which is then used as a normalization. Location is approximated both using unsupervised geometric discretization, and supervised administrative hierarchies, which permits detecting events at city, regional, and global levels at the same time.

In addition to these main contributions we will provide discussion about the similarities of event detection and Outlier Detection. Further we will show how complex algorithms such as the classification of trends based on their dissimilation patterns or Geo-spatial co-location mining can benefit from Event Detection as a preprocessing step or become feasible at all. Furthermore, we demonstrate the accuracy of our methods against a controlled dataflow with artificially injected events and illustrate the effectiveness with real world examples such as reporting of significant earthquakes, detecting names of football players during famous soccer matches and in studying New Year celebrations around the world.

Zusammenfassung

Social-Media-Plattformen wie Twitter oder Facebook sind eine beliebte Quelle für Echtzeit-Textdaten. Zusammen mit täglich veröffentlichten Artikeln aus Zeitungen und Web-Blogs bilden sie eine überwältigende Menge an Daten. Diese Flut macht es Menschen unmöglich, abgesehen von populären Schlagzeilen, alle Informationen zu verarbeiten.

Durch Ereignis-Erkennung könnten diese Daten in Trend-Themen zusammengefasst werden und somit besser von Benutzern erfasst werden. Doch solche Daten zu analysieren ist eine Herausforderung. Algorithmen müssen mit großen Datenmengen umgehen und sich schnell anpassen können. Datensätze treffen mit hoher Frequenz ein und sind äußerst heterogen, da sie aus Text, geografischen Standorten und Bildern bestehen. Aufgrund der wachsenden Nachfrage von Methoden zur Zusammenfassung und Abstraktion gepaart mit der zunehmenden Verfügbarkeit von öffentlichen Text Datenquellen erfuhr die Trend- und Ereignis-Erkennung erhöhte Aufmerksamkeit, sowohl seitens der Forschung als auch der Industrie.

Bestehende Methoden sind allerdings oft stark in ihrem Funktionsumfang beschränkt: Sie arbeiten nur mit vom Benutzer vorgegebenen Schlagworten, analysieren nur eine kleine Teilmenge (wie beispielsweise Hashtags) oder sind lediglich in der Lage, globale Ereignisse (wie den NFL Super Bowl oder Naturkatastrophen) zu erkennen.

In dieser Arbeit werden wir mit unserer Forschung zu dem Gebiet der Ereignis-Erkennung beitragen. Als ersten Beitrag stellen wir eine Methode vor, die in der Lage ist, große Datenmengen zu verarbeiten. Dies geschieht durch einen effizienten inkrementellen Algorithmus, der ungewöhnliche Textmuster im Datenstrom identifiziert. Der Benutzer braucht hierfür keinerlei Einschränkungen bezüglich der Daten zu spezifizieren (wie z.B. Hashtags). Dazu stellen wir ein Signifikanzmaß vor, um Ereignisse zu erkennen, bevor sie zu Schlagzeilen werden. Mit Methoden aus dem Forschungsgebiet der Ausreißererkennung und mit Hash-Tabellen, ähnlich der Heavy-Hitter Datenstruktur, werden wir einen Algorithmus entwerfen, der alle Worte, Wort Paare und Positionsdaten effizient verarbeiten kann. Durch einen Vergleich neu beobachteter Werte mit Statistiken der früheren Daten, können wir sofort signifikante Abweichungen mit minimaler Verzögerung melden. Dies macht unser System zu einer vollautomatischen sozialen Medien-Berichterstattung in Echtzeit.

Als zweiten Beitrag befassen wir uns mit der ungleich verteilten Nutzung von Social Media, welche zu verzerrten Ergebnissen führt, falls diese nicht sorgfältig geprüft wird. Beispielsweise trägt London mit einem Anteil von 0,67% zum gesamten Twitter Volumen bei. Dies ist mehr als doppelt so viel wie der Anteil Deutschlands mit lediglich 0,31%. Durch die Nutzung der mit Positionsdaten angereicherten Texte können wir die lokale Dichte einer jeden geographischen Region schätzen. Dadurch sind wir in der Lage, diese Ungleichheiten zu normalisieren und lokal signifikante Ereignisse zu erkennen.

Zusätzlich zu diesen beiden Hauptbeiträgen werden wir eine Diskussion über die Ähnlichkeiten von Ereignis-Erkennung und Ausreißer-Erkennung führen. Weiterhin zeigen wir, wie komplexe Algorithmen von der Ereignis-Erkennung als Vorverarbeitungsschritt profitieren können oder sogar dadurch erst machbar werden. Erkannte Ereignisse können beispielsweise anhand ihrer geographischen Ausbreitungsmuster klassifiziert werden, um ihre weitere Ausbreitung vorherzusagen. Desweiteren werden wir zeigen, wie wir soziale Gruppen finden, die häufig die selben Standorte besuchen und wie wir Ereignisse nutzen um eine soziale Landkarte zu erstellen, welche in ähnliche Themenbereiche gruppiert ist. Darüber hinaus zeigen wir die Korrektheit unserer Methoden anhand eines Datensatzes mit künstlich injizierten Ereignissen und demonstrieren die Wirksamkeit unseres Algorithmus anhand realer Beispiele wie die Erkennung von Erdbeben oder durch detaillierte Analyse einzelner Großereignisse wie die Fußball Weltmeisterschaft oder das Neujahrsfest.

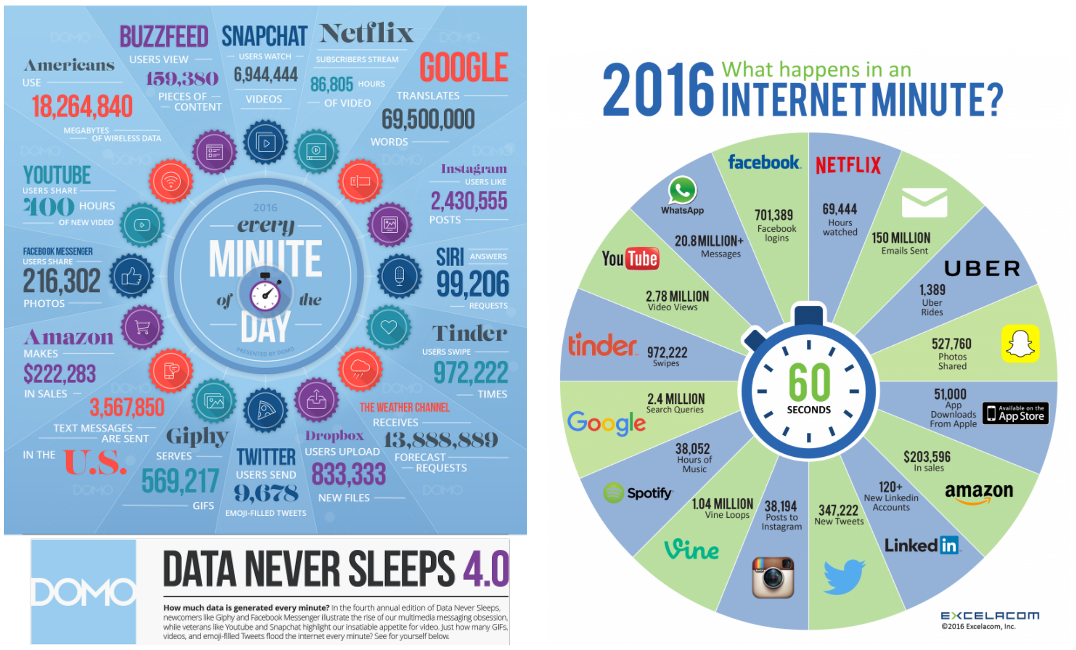
Part I

Preface

Chapter 1

Introduction

Live textual data from social media, blogs and daily news produce an overwhelming amount of data. Many reports exist about this ever increasing data. In Figure 1.1 you can see how much data is generated on average in the year 2016: Twitter users send about 9.678 Tweets with emojis or 159, 380 pieces of content are created at BuzzFeed[1] every minute. Social networks and microblogging services such as Twitter, gained a lot of interest due to its large amount of data published every second by their users. Furthermore, there are a number of global events – such as the Arab Spring – where Twitter is reported to have had a major influence [80]. But also news agencies like Reuters and Bloomberg



(a) Data Never Sleeps 4.0 (b) What happens in one Internet Minute

Figure 1.1: Blog posts from Domo [1] and Excelacom [2] how much data is generated per minute in the year 2016.

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

publish thousands of articles daily covering a wide range of topics. This flood of information makes it infeasible for individuals to keep the overview beyond a certain level of popularity, like the global top stories of popular news papers. This information explosion calls for new tools and approaches to process this amount of data, as a single user can no longer read all the information available. Instead, automatic filters are used everywhere: email servers reject two thirds of our email traffic deemed as spam; the Facebook news feed is also heavily filtered: Facebook reported that the average user would have 1,500 potential stories each day, out of which around 300 are automatically prioritized and shown by Facebook; yet most users only read about half of these on average.[2] In addition, some hard coded filters are used to e.g. merge all birthday wishes into a single story to reduce clutter.

By detecting emerging topics, data could be summarised by its corresponding series of events, which makes it easier for users to understand what actually happens regarding multiple topics at the same time. But analysing such data is challenging: Algorithms need to handle very large, rapidly changing and fast arriving data sets that are heterogeneous, consisting of text, geographic locations and images. Due to the growing need for summarization and abstraction combined with the increasing availability of public textual data sources, the field of emerging topics detection has recently gained a lot of attraction in research as well as in industry.

However, existing methods for emerging topic detection are often only able to detect events of a global magnitude such as the NFL Super Bowl, natural disasters or celebrity deaths. Furthermore, existing methods are often limited in what they track: They require a user to specify a set of keywords in advance or operate only on a limited subset like hashtags. Interesting emerging topics may, however, be of much smaller magnitude and may involve the combination of two or more words that themselves are not unusually hot at that time.

In Section 9 we will discuss that this Event Detection process is closely related to the field of Outlier Detection. We expand a generalized framework for Outlier Detection proposed by Schubert et al. [170] to also cover *trend detection* and *emerging topic detection*.

With the event detection method proposed in Part II we are able to track *all* words, *all* locations and even *all* pairs of co-occurring words simultaneously. This means that we do *not* need to restrict the set of words or locations beforehand and thus are able to use Event Detection as a preprocessing step.

Algorithms that would not be feasible on the raw input data could instead work solely on the remaining outliers (trends and events) and thus be much more efficient; or just become feasible at all. For best flexibility of our system design (such as the one shown in Figure 1.2), we need to make sure that we minimize the probability to discard information that could be potentially important to an algorithm further down our processing pipeline. For example,

[2] `http://www.facebook.com/business/news/News-Feed-FYI-A-Window-Into-News-Feed`

a sentiment analysis algorithm would perform badly when its input data consists solely of popular hashtags, as most sentiment analysis algorithms rely on emoticons to perform the classification of sentiments correctly [97]. As we do not suffer from such restrictions we will be able to use our Event Detection as a decoupled microservice in a distributed larger system and enable a large variety of algorithms to work on the remaining event candidates efficiently. In Part III we will discuss further algorithms in detail that use events as input to

- classify them based on their spatial dissemination features (see Chapter 11),
- build a Socio Textual map (see Chapter 13) that reveals groups with similar interests and thoughts, and
- use co-location mining of events (see Chapter 12) to identify social groups that are frequently found at the same location.

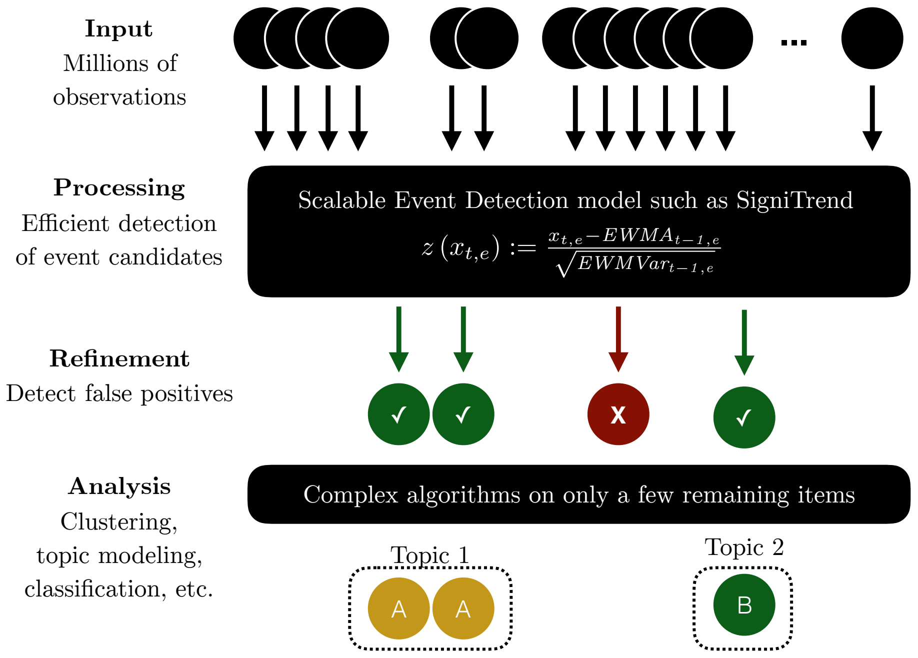

Input
Millions of observations
Processing
Efficient detection of event candidates
Scalable Event Detection model such as SigniTrend
$z(x_{t,e}) := \frac{x_{t,e} - EWMA_{t-1,e}}{\sqrt{EWMVar_{t-1,e}}}$
Refinement
Detect false positives
✓
✓
X
✓
Analysis
Clustering, topic modeling, classification, etc.
Complex algorithms on only a few remaining items
Topic 1
A
A
Topic 2
B

Figure 1.2: Event detection as preprocessing filter step for further analysis and expensive algorithms.

Chapter 2

Thesis Overview and Contributions

This chapter gives an overview of this thesis and highlights the contributions of publications used in this thesis. A more detailed discussion about individual contributions could be found in the chapters of the corresponding publications.

2.1 Trend Mining and Event Detection

In Part II we will introduce the publications "SigniTrend: Scalable Detection of Emerging Topics in Textual Streams by Hashed Significance Thresholds" [163], "SpotHot: Scalable Detection of Geo-spatial Events in Large Textual Streams" [166], "Scalable Detection of Emerging Topics and Geo-spatial Events in Large Textual Streams" [165] and "Outlier Detection and Trend Detection: Two Sides of the Same Coin" [167]. Our contributions to the field of emerging topics and event detection are as follows.

Significance Measure: We propose a significance measure that can be used to detect emerging topics early, long before they become "hot topics", by drawing upon experience from Outlier Detection.

Probabilistic Counting: By using hash tables in a heavy-hitters type algorithm for establishing a noise baseline, we show how to track even all word pairs using only a fixed amount of memory.

Aggregation into Topics: We aggregate the detected co-trends into larger topics using clustering approaches, as often as a single event will cause multiple word combinations to trend at the same time.

Geo Normalization and Location Modeling: We also address the problems arising from differences in adoption of social media across cultures, languages, and countries by efficient normalization. Therefore, location is modeled using unsupervised geometric discretization and supervised administrative hierarchies, which permits detecting events at city, regional, and global levels at the same time.

Relationship to Outlier-Detection: To bridge the gap between events and outliers we show their relationship and propose an extension of a generalized framework for Outlier Detection to also cover *emerging topic detection.*

2.2 Geo-social Co-location Mining of Events

In Part III we introduce advanced event analytic methods that use our Trend Mining and Event Detection from Part II as a “filtering” step. Thus we are left with only a small fraction of words that now represent topics and events. In the following, we describe our contributions based on the publications “Geo-social Co-location Mining” [193], “Socio Textual Mapping” [194] and “TrendTracker: Modelling the motion of trends in space and time”[161].

Trend Archetypes: We postulate and verify a hypothesis that trends follow different archetypes, which differ strongly in terms of their dissemination patterns. Using a clustering approach, we identify these archetypes and thus are able to classify newly discovered trends based on their corresponding archetype features.

Geo-social Co-location Mining: For the problem of finding social groups that are frequently found at the same location, we propose a probabilistic model. With this model we are able to estimate the probability of a user to be located at a given location at a given time. We extend solutions for probabilistic frequent itemset mining to be able to process for large amounts of data that have a high degree of uncertainty.

Geo-social Map: We define a dissimilarity measure and utilize a metric clustering approach to obtain a social map of regions with similar events. We formalized this idea as a *vision* towards a semantical clustering approach and provide an initial proof of concept.

Chapter 3

Preliminaries

This chapter will give a brief introduction into basic concepts that are used in this thesis like hashing Bloom Filters and natural language processing methods. This is *not* meant to neither be an comprehensive nor complete coverage of the following topics. The purpose lies solely in making the reader familiar with the relevant methods and concepts. When discussing the preliminaries we will have a strong focus on their actual use case and benefits regarding the challenges in this thesis.

3.1 Hashing

As we will later see in Part II, hashing will build our algorithm core to speed up the trend and event detection computations.

Hash Function: A hash function can map data from a large domain of arbitrary size to a smaller domain with fixed size. A simple hash function for strings (text) could be a mapping from a string to its first character. Such a function would produce the following mappings for the strings "foo", "bar" and "baz" as shown in Table 3.1.

Hash Key	Corresponding Value
f	foo
b	bar, baz

Table 3.1: Example mapping of a simple hash function for strings.

Hash functions are applied to a variety of different use cases like quick lookups for HashTable data structures (also called "Dictionaries") or within the field of cryptography. A real example of a hash function could be found within the implementation of the method `hashCode()` from Java (see `java.lang.String` implementation in Java SE[1]). For a given string s, the corresponding hash

[1]`http://docs.oracle.com/javase/1.5.0/docs/api/java/lang/Object.html#hashCode`

function is there defined as follows.

$$h(s) = \sum_{i=0}^{n-1} s\,[\,i\,] \cdot 31^{n-1-i}$$

Where $s\,[\,i\,]$ denotes the ordinal value of the i-th character and n the total number of characters of the string s.

Hash Collisions: As shown in Table 3.1, different input values can produce the same hash value. Most applications need to carefully handle this cases of collisions to not produce incorrect results. As we will later see in Part II, we design our trend and event detection algorithms with collisions in mind. When tracking statistics of current words from high throughput text data streams, we work with those collisions: We accept them to happen frequently and design our calculations with additional strategies to guarantee a minimum error with upper and lower boundaries. Additionally we will see how we can ensure that the error gets lower when occurrence frequency of words gets higher. This means that we could track events (that are not rare by definition) with minimal differences to their actual values.

3.2 Bloom Filter

A Bloom filter [38] is a data structure to test set-membership of elements very efficiently regarding both, memory space and computation time. This performance comes at a price: Bloom filter membership tests can only be executed in a probabilistic manner, so that false positive matches are possible. This means that a Bloom filter can answer a membership query either with "No!" or "Maybe?!". Furthermore, a Bloom filter does *not* actually store the elements, thus, one cannot receive all "inserted" elements.

A Bloom filter consists of a bit array of b elements (also called buckets), which are initially set to 0. In the following, we call this parameter b the *bucket size* of our Bloom filter. When elements are added, k different hash values $H = \{h_1, \ldots, h_k\}$ are being calculated, where each h_i represents a hash function that maps elements from the input domain to an index $i \in [0; b[$ to address one of the Bloom filters buckets. For answering membership queries we simply calculate the k hash indices of the query element. If we then find a 0 among the buckets we can respond with "false", as the query element can definitively not be in our set (because it previously would have caused to flip *all* those bits to 1).

In Figure 3.1 we can see a small Bloom filter example with $k = 2$ hash functions and a bucket size $b = 8$. When inserting words like "Apple", "Banana" and "Cherry" we flip all bits corresponding to their hash index to 1. A query for "Blackberry" would result in a true negative membership test, since at least one hash does not collide with any other hash (here only one hash collides with one hash of the word "Banana"). However, a membership query for the word

"Blueberry" would produce a false positive match, since all of its hashes collide with at least one of the hashes (here with "Apple" and "Cherry").

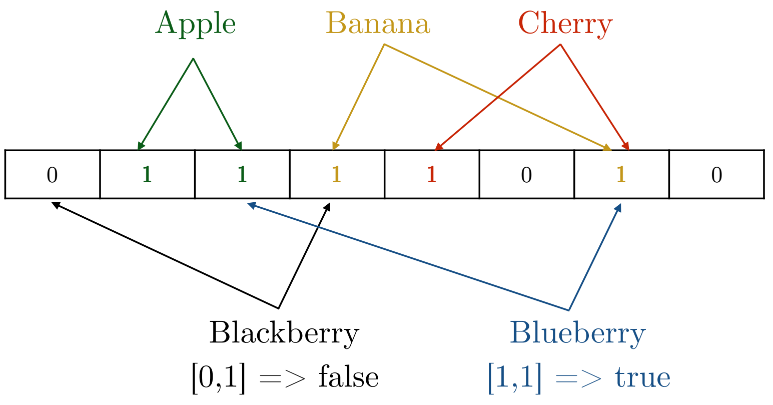


Figure 3.1: Example usage of a Bloom filter with $k = 2$ hash functions.

3.3 Count–Min Sketch

The Count-Min Sketch, invented 2003 by Cormode and Muthukrishnan [58], is a data structure that stores the frequency of an observation. Similar to the Bloom filter, the counting of the frequencies is done in a probabilistic manner for maximum efficiency. This means that unlike an associative array (such as the Java HashMap[2]) where each key is mapped to a certain value[3], the Count-min Sketch uses hash functions to determine the corresponding buckets and updates the frequencies according to the corresponding buckets. As this can cause collisions, the Count-min sketch will eventually overestimate certain input values. In Figure 3.2 an example Count-Min sketch is illustrated with $k = 2$ hash functions and a bucket size of $b = 8$ buckets similar to the Bloom filter discussed in Section 3.2. When the words "Apple", "Banana" and "Cherry" are inserted we will see that the two observations of "Apple" and "Cherry" will override the single observation "Banana" at the penultimate bucket. Due to such collisions, a frequency lookup could result in an overestimation: For instance, a lookup for the word "Banana" would result in an overestimated frequency of 2 (instead of its real frequency 1), since 2 is the minimum value of all hash bucket collisions with "Apple" (frequency 3) and "Cherry" (frequency 2). False positives can still be produced as in the example of the lookup for "Blueberry": As this was *not* inserted the frequency should be 0; instead we see an overestimation of 2 (minimum of 3 from "Apple" and 2 from "Cherry").

[2]`https://docs.oracle.com/javase/8/docs/api/java/util/HashMap.html`

[3]with a specific strategy to handle hash collisions (such as buckets with associated lists of overflow entries)

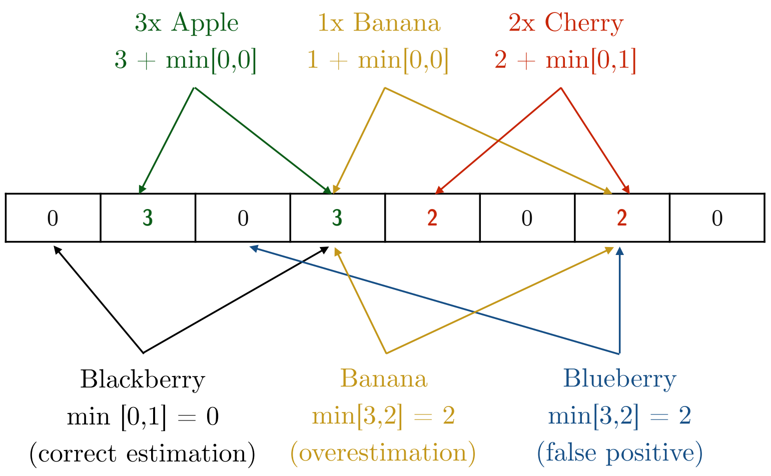


Figure 3.2: Example usage of a Count-Min sketch with $k = 2$ hash functions. Frequencies can be overestimated and false positives are still possible.

3.4 Zipf's Law

Zipf's law is an empirical formulation for the fact that many types of data in the field of social sciences can be approximated with a Zipfian distribution. For N number of elements, Zipf's law predicts the frequency of an element with rank k as follows:

$$f(k, s, N) = \frac{1/k^s}{\sum_{n=1}^{N}(1/n^s)},$$

where s denotes the value of the exponent that characterizes the distribution. In natural languages like English or German, word frequencies have a long-tailed distribution. Such word frequency distribution can be modeled by a Zipf distribution with $s \approx 1$ [122]. This means that there are only a few words with very high frequencies and a lot of words that appear rarely. In Twitter this holds in particularly as there are a lot of spelling errors and abbreviations. Additionally users often include other user names (the so called "user mentions") within their tweets to talk about other users. All this leads to a large amount of distinct vocabulary because there are a lot of ways how words can be misspelled. Likewise the majority of user names will appear rarely in other tweets (except the names of some popular users).

In Figure 3.3 we show the frequency distribution of a typical week of tweets.

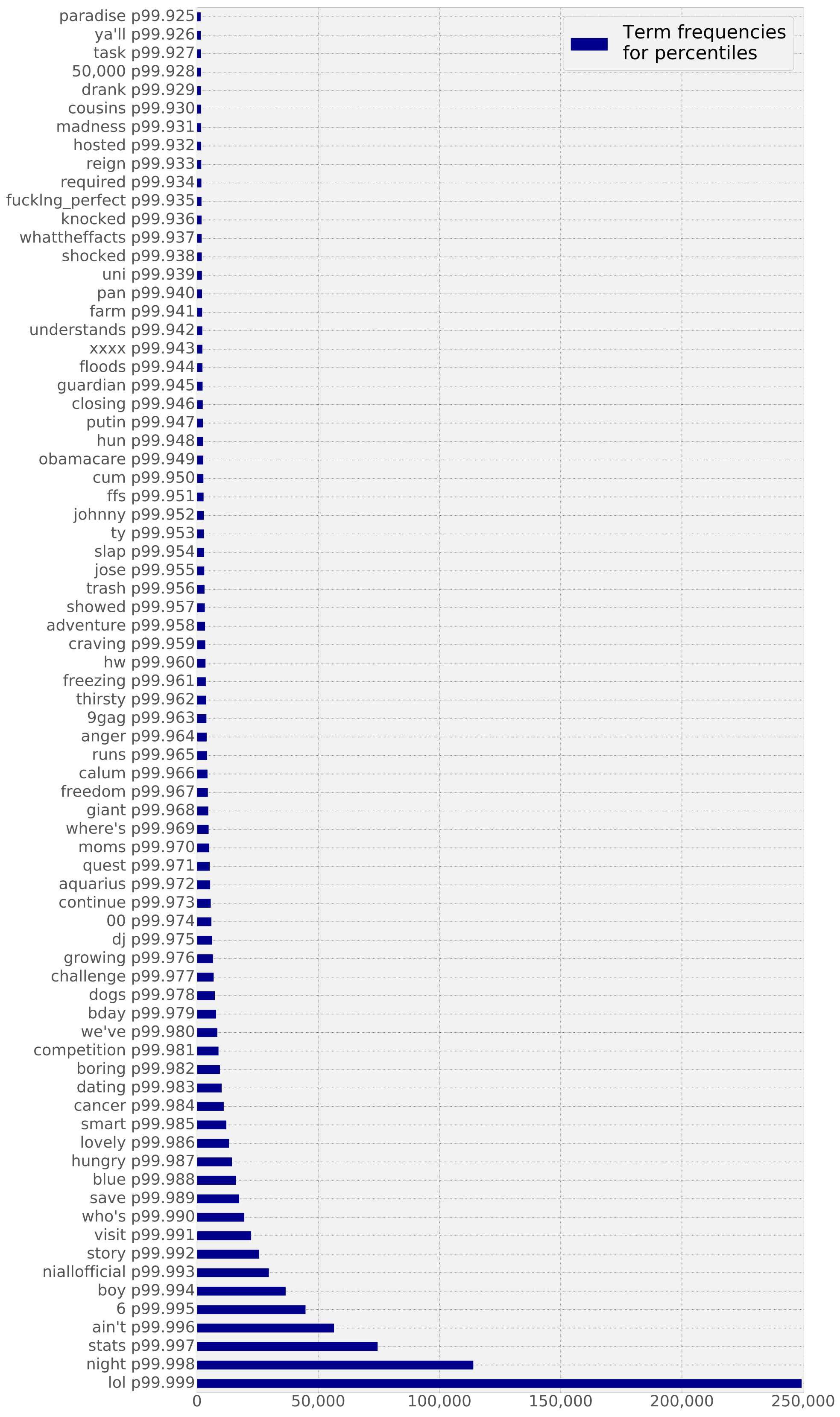


Figure 3.3: Zipf-like long-tail distribution for term frequencies for one typical week of our tweet data set. X-axis: the occurrence frequency. Y-axis: Percentiles with their corresponding words.

Table 3.2: Term frequency statistics for one typical week of English tweets in our data set.

Description	Value
Number of terms	127,976,527
Number of distinct terms (vocabulary)	6,452,098
Number of terms that appeared only once	4,978,292
Maximum term frequency	3,774,815
Standard deviation of term frequency	3,625
Mean term frequency	19.8
Median term frequency	1

To extract word tokens we used a standard *Lucene Tokenizer* (see Section 3.6 for more details on tokenization). As you can see, only few terms – mostly stop words with low information – such as "lol", "night" or "ain't" contribute to the high frequencies within the top 99.9% percentile. The row "lol p99.999" for example shows that the term "lol" is ranked at position 99.999% in a list of words (sorted ascendingly by their occurrence frequency). For visualization reasons we excluded the top most frequent term "rt" (that would refer to percentile $p = 100\%$). The term "rt" is especially common for Twitter as it is related to the Twitter specific abbreviation for denoting a re-tweet. The long-tail of this distribution is built by words with low frequencies. Even within our small excerpt from percentiles 99.9% to 99.8% the frequencies drop very fast. Further statistics could be obtained from Table 3.2: from the total number of 127,976,527 observed terms a vocabulary consisting of 6,452,098 distinct terms was built. A large fraction of ≈ 77% of this vocabulary appeared only a single time within this one week.

3.5 Term Frequency–Inverse Document Frequency

The term frequency–inverse document frequency or short *tf-idf* is a concept from information retrieval to measure how important a word is to a document. Tf-idf is based on the heuristic, that rare words contribute more information to a document in which they appear often. As a consequence, frequent words like "the", "a" and "this" will get low scores in typical English documents as they are frequently used in almost all documents. For a given set of documents D, the tf-idf score for a term t and a document $d \in D$ is given by a product of the term frequency tf and the inverse document frequency idf.

$$\mathrm{tfidf}_{t,d,D} = \mathrm{tf}_{t,d} \cdot \mathrm{idf}_{t,D}$$

The term frequency $\mathrm{tf}_{t,d}$ measures how important a term t is for a document $d \in D$, whereas the inverse document frequency $\mathrm{idf}_{t,D}$ measures how important a term is within the complete document corpus D. A simple implementation of $\mathrm{tf}_{t,d}$ uses the raw number of occurrences of term t within document d. Let $N = |D|$ denote the number of documents in our document corpus D. The idf score of term t and document d is then defined as follows:

$$\mathrm{idf}_{t,D} = \log \frac{N}{|\{d \in D : t \in d\}|}$$

where $|\{d \in D : t \in d\}|$ denotes the number of all documents where the term t appears. Many other variants of both tf and idf have been established since tf-idf was introduced by Luhn [127] in the year 1957. For the scope of this thesis, the aforementioned definitions are sufficient. For more details and theoretical arguments regarding this topic see [153] and [159].

As we will discuss in Section 5.4, we cannot use tf-idf directly, because tf-idf was designed to quantify the similarity of two documents. Instead our Event Detection system can benefit from such term weighting *after* words were detected as part of an event.

3.6 Stemming and Tokenization

Tokenization is the process of splitting an input text into chunks that represent meaningful entities such as words and punctuation. Throughout this thesis we use the notions "word", "term" and "token" interchangeable with each other. To produce a list of tokens we make use of the method `tokenStream()` of the class `StandardAnalyzer` class[4] that is part of the open source information retrieval software library Apache Lucene. The following Java code snippet will show an example usage:

```java
TokenStream stream = new StandardAnalyzer().tokenStream(
    ↪ null, new StringReader("hello,␣world!"));
stream.reset();
while (stream.incrementToken()) {
    String token = stream.getAttribute(
        ↪ CharTermAttribute.class).toString();
}
```

A commonly used preprocessing step is the reduction of tokens to their stem (or root) form (e.g. "running" and "runs" are each mapped to their base form "run"). In information retrieval this process is called *stemming*. The stem itself does not need to be a valid root (e.g. "argue", "argued" and "argues" can be reduced to "argu"). In Part II we will apply stemming to be able to generate more stable statistics while reducing the size of our vocabulary. For the preprocessing step in our Event Detection we are making use of the Xapian search engine library[5] that provides language specific stemming methods.

[4]`https://lucene.apache.org/core/5_2_0/analyzers-common/org/apache/lucene/analysis/standard/StandardAnalyzer.html`

[5]Open-Source, `http://xapian.org/`

3.7 Outliers

Intuitively, outliers could be described as elements that are unusual observations within a data set. Depending on the use case outliers could be the object of interest (e.g. detect credit card frauds among transactions). In other scenarios, outliers are treated as noise that will get removed from a data set in a preprocessing phase (e.g. measurement errors or experimental errors).

Outliers are hard – if not impossible – to define mathematically because we cannot expect them to follow a model or distribution known beforehand. Instead, most attempts at defining outliers focus on them being a rare observation, markedly different from the remainder of the data, such as the well-known definition by Barnett, Toby and Grubbs:

> "an observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data".
> — *Vic Barnett and Toby Lewis* [32]

> An outlying observation, or "outlier," is one that appears to deviate markedly from other members of the sample in which it occurs.
> — *Frank E. Grubbs* [75]

Yet, in each definition, the notions of "inconsistence" and "remainder" remain vague. Various algorithms have been proposed that try to detect outliers in a way consistent with our intuition. Notable Outlier Detection algorithms include DB-Outlier [94], which reports the objects of lowest density as outliers and local outlier factor (LOF) [39] which uses a local neighborhood as "remainder" of the data set, instead of comparing to the complete data set every time. The basic idea is to assign a local density estimate (local reachability density, *lrd*) to each object of the database. Then, LOF considers ratios between the *lrd* of an object and the *lrd*s of its neighboring objects. Thus, the resulting outlier score is based on a local comparison rather than on a global comparison. In Figure 3.4 you can see a visualization of the basic idea of LOF: the local density of a point gets compared with the densities of its neighbors. Outliers, such as the data point marked as "X", shows a much lower density than its neighbors "A", "B" and "C".

Schubert et al. [170] propose a generalized framework for Outlier Detection that is applicable beyond the domain of vector spaces and show the applicability to graph and video data. In Figure 3.4 you can see the result of their Outlier Detection method. They analysed the unemployment rate of Germany. Cities were detected as outliers as they usually show significantly higher unemployment rates than their sparse populated neighborhood. In Chapter 9 we will discuss how the field of Outlier Detection is closely related to the field of event detection, as both try to find unusual, rare and abnormal observations.

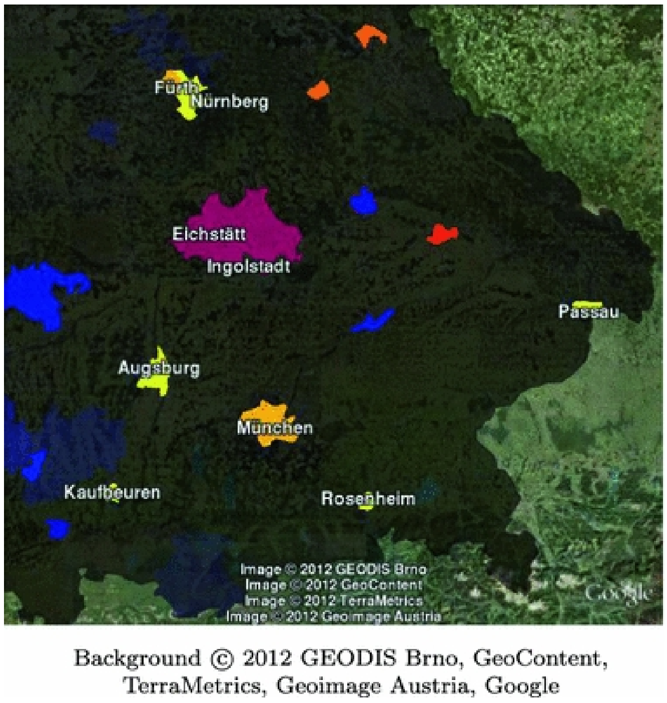

Fürth
Nürnberg
Eichstätt
Ingolstadt
Passau
Augsburg
München
Kaufbeuren
Rosenheim
Image © 2012 GEODIS Brno
Image © 2012 GeoContent
Image © 2012 TerraMetrics
Image © 2012 Geoimage Austria
Google
Background © 2012 GEODIS Brno, GeoContent, TerraMetrics, Geoimage Austria, Google

Figure 3.4: Results for LOF on unemployment rates of Bavaria (Germany). Cities up as outliers, as they usually have a significantly higher unemployment rate than the sparse populated surrounding areas. (Figure obtained from [170])

3.8 Trends, Events and Emerging Topics

In the scope of this thesis we will use the terms "trend", "event" and "emerging topic" interchangeably. Colloquial, the word "trend" often describes an increase of interest over a longer time period. This ranges from fashion trends to computer science trends such as big data and distributed computing. Typically no specific point in time is associated to those; interest may start small and then grows continuously. Identifying the source that mentioned a trend *first* is often non-trivial. Big data, for example, was getting much attention after the report "Big data: The next frontier for innovation, competition, and productivity" [130] by McKinsey in 2011. However, Big Data was described earlier in a research report from Doug Laney, an analyst of the META Group (now Gartner) in 2001. In his article "3D data management: Controlling data volume, velocity and variety" [107] he introduced the 3 V's (volume, velocity and variety). But the history of Big Data goes back even further: the Forbes article "A Very Short History Of Big Data"[150] describes that the first mentions regarding the growth rate and the volume of data ranges back to the year 1944 when the librarian Fremont Rider published his report "Scholar and the Future of the Research Library"[152] where he estimated that "American university libraries were doubling in size every sixteen years". And likewise the International Conference on Very Large Databases (VLDB): Since 1975 researchers published their work targeting large data sets and data management.

Due to such ambiguousness this thesis does not focus on detecting "the first" source and instead refers to the research field of First Story Detection (FSD) [146, 19]. Further, we measure interest quantitatively with the number of mentions from a data source such as Twitter or a news agency. This means that we do not determine qualitative factors such as sentiments of an event, but design our system to enable such analysis at a later step (as shown in Figure 1.2).

Closely related to trends and events is the term "hype", as it is also used to describe topics of increasing interest. One famous source for technology hypes is the *Hype Cycle*, a report that is released regularly by Gartner, Inc. As you can see in Figure 3.5, the term "Big Data" is placed on the 3rd slice called "Trough of Disillusionment" which means that the interest has reached its peak and is now declining.

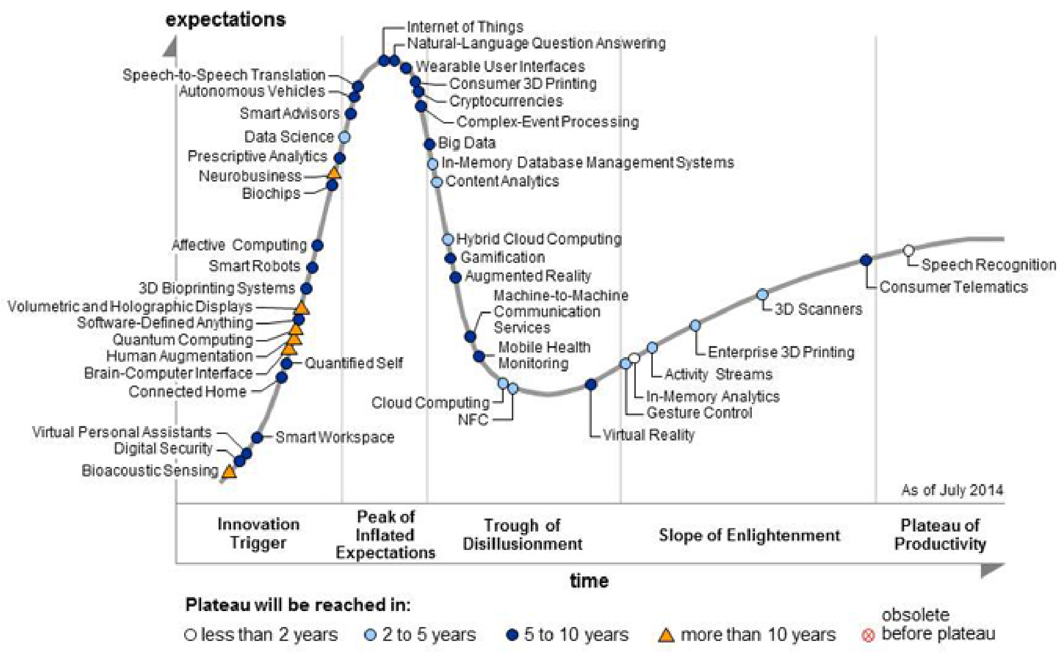


Figure 3.5: Hype Cycle for Emerging Technologies, 2014. "Big Data" is placed on the 3rd slice "Trough of Disillusionment" which means that the interest has reached its peak and is now declining. Source: Gartner, Inc.

3.9 Gold Standard

A lot of data sets for anomaly detection in time series exist. In the case of textual events they are often not suited as a baseline as they have specific characteristics and definitions of outlierness. Consider the case of a heart rate analysis from Mateo et al. [131] on electrocardiography (ECG) data. If we use a our Event Detection algorithm – that measures how many standard deviations the current signal differs from its moving average – on the ECG data shown in Figure 3.6 we would detect the first peak of each as an event.

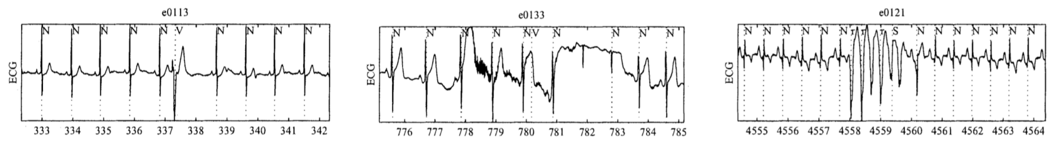


Figure 3.6: ECG anomaly detection from Mateo et al. [131].

As both, the standard deviation and the moving average are now increased, we would suppress the subsequent peaks and no event alert is reported. While this is comprehensible for textual data our algorithm would perform badly for this special purpose of ECG data analysis, where small irregularities in the frequency pattern can have a strong impact on the patients health.

Additionally when time series data is produced by high precision sensors, the detection of events could be much more sensitive and react immediately to minimal deviations. Again, text behaves differently: Spelling errors and ambiguous words exist. But most importantly users from social media do not behave like high precision sensors[6]: a single user may write a lot of consecutive messages just because he or she is waiting for something or has found a fast Wi-Fi network while on holiday. In the case of (large) social networks we observe millions of users simultaneously. Thus, we need to compensate for such random behaviour by aggregating observations to statistically stable chunks. If we instead would naively adjust our Event Detection parameters to be able to report minimal deviations we would produce an infeasible amount of event reports, whereas most of them would correspond to random behaviour and not real trends. The problem that arises when multiple statistical inferences are considered simultaneously is called “multiple testing problem” [155].

Further, as previously discussed in Section 3.11.2, normalization can distort data sets. Temporary peaks can thus get extinguished by consecutive slices with low frequency. The question whether or not it is correct to classify a significant frequency increase within a small time slice (say, 1 minute) as an event, even if it is not significant when data is aggregated into larger slices (say 1 hour), is clearly dependent on the user’s perspective.

In conclusion, textual data from social networks behaves different then high precision sensors and because we observe millions of users at the same time we have a high chance to find random patterns when looking at a aggregation over a too small time slice. Thus, we provide parameters such as the *epoch size* and *ageing* to control the aggregation size as well as the amount of data that gets wiped out to enable recurrent events and give the user control over the sensitivity of the Event Detection system. Further, this thesis focuses on

[6]Even though there are existing efforts from the research community, e.g. Sakaki et al. [157] to threat Twitter users as sensor data

detecting events with positive interest peaks and does not consider the absence of an expected peak as an event. This means that we do not detect irregularities regarding patterns, such as “Monday” was mentioned less than expected for typical Monday or that users stopped talking about “Michael Jackson”.

3.10 Twitter

Twitter is on of the most famous online social networking services. Users can publish a short fragment of text (up to 140 characters), some annotated entities and links to other users as well as web sites. Additionally users can add location information on where the message originated from. In many cases Twitter is used as primary source due to its free and easy to access streaming API.

For most of the experiments in this thesis we use Twitter as the main source of textual as well as location data. The `statuses/sample` endpoint of Twitter includes 1% of all tweets (4–5 million per day, along with 1–2 million deletions; about 15 GB of uncompressed raw data per day). A surprisingly small percentage of 1.5% of these tweets include geographic coordinates: Retweets never contain coordinates, and often users only check in with an approximate location such as a point of interest. Assuming the sample is random, the estimated volume is 7.5 million tweets per day with coordinates.

3.11 Time Series Data

In this section we discuss event detection on one-dimensional time series data. We use data from publicly available sources to show how frequency bursts can be leveled out by increasing aggregation window duration. Also we discuss the necessity of normalization to not draw wrong conclusions on current observations.

3.11.1 The Importance of Aggregation

Without aggregation analysing time series data would yield a lot of unstable frequencies. Additionally, most real world data sets have an implicit aggregation interval due to technical limitations such as sensor interval rates. In the case of Twitter data for example frequencies less than 1 second barely make sense due to network latencies and server side micro-accumulation.

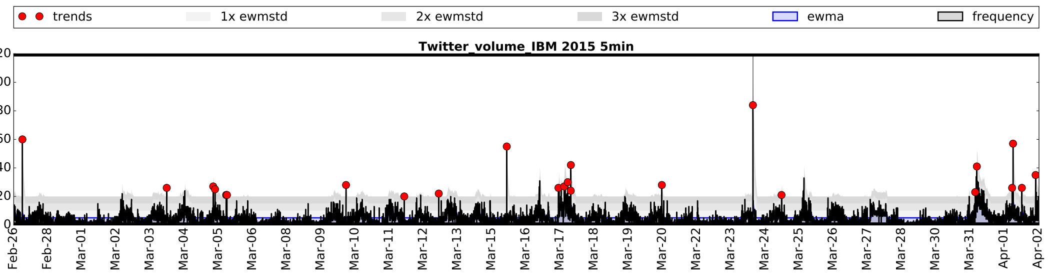
To demonstrate the impact of aggregation we use data provided by the *Numenta Anomaly Benchmark* [111] available at `https://github.com/numenta/NAB`. In Figure 3.7 we can see the number of mentions of the stock symbol *IBM* on Twitter on different aggregation levels. In Figure 3.7a the finest aggregation with a 5 minute window is shown. The ground truth for points in time where anomalies occur are also given within this data set. The bucket which contains the anomaly is thereby marked in green. Points in time with frequencies that are detected as “events” by the obtained z-score of SigniTrend (detailed theory in Section 5.4) are highlighted with red dots. When the aggregation interval is increased from 5 minutes to 1 hour (Figure 3.7b) the curve becomes smoother and a lot of the frequency bursts are now flattened as you can see in the reduced amount of red dots.

When the aggregation reaches window sizes of 12 hours (Figure 3.7c) and 24 hours (Figure 3.7d) only two significant detected events remain. Note that the bucket at March 23th, containing the manually labeled anomaly, disappeared and is not detected as trend anymore. This effect is of course present in a lot of data sets and comes in a variety of scales: Users talk daily about eating "breakfast" at morning or wishing each other a "good night" in the evening. Every week tips for spending the "weekend" are discussed. Such patterns could also be observed with annually recurring events such as Valentine's Day or April Fools Day. These days show significant peeks at that day but would vanish if data gets aggregated into epoch sizes of one complete year.

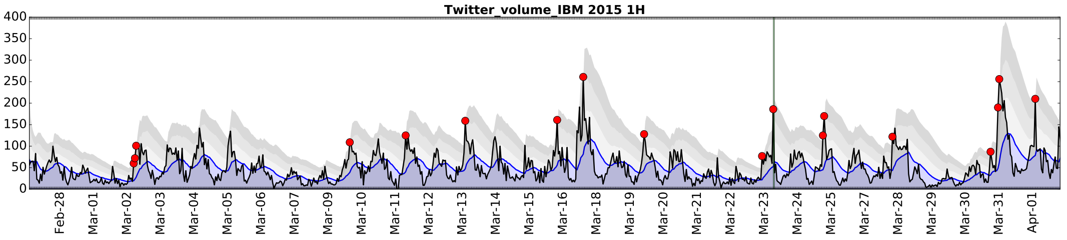
Whether or not those recurring events are "real" trends depends on the user's perspective. As we will later show in Chapter 5 our trend detection is capable of handling different aggregation levels with an exact-counting mini batch approach and gives the user the choice to handle recurring events with two simple parameters.

3.11.2 Normalization

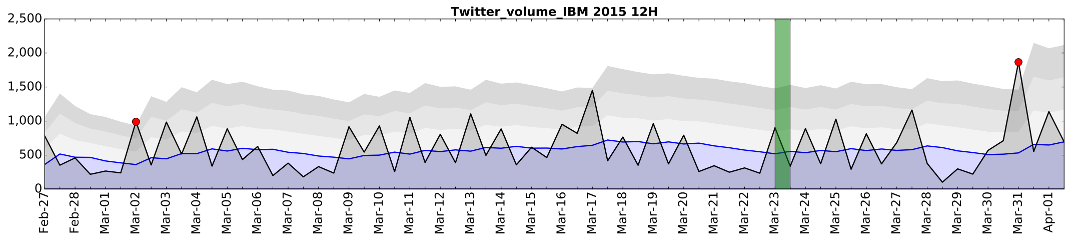
Over time, the volume of observations can vary heavily. Especially data sources that involve the actions of users show high fluctuation in their overall volume due to day and night cycles. In Figure 3.8 we visualized the total amount of observed tweets and those that contain text that is classified as English from Twitter itself. This can cause uninteresting words to show an increase in their frequency although its occurrence probability would remain stable. Furthermore, the absolute popularity is subject to seasonal trends. Depending on the data source, working hours and weekdays would cause "seasonal" patterns that need to be accounted for. Relative popularity – normalized by the total number of documents for the day – proved much more robust in our experiments.



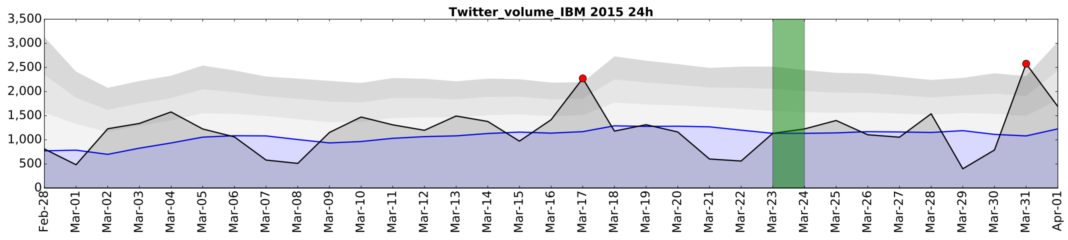
(a) 5 minutes aggregation



(b) 1 hour aggregation



(c) 12 hour aggregation



(d) 1 day aggregation

Figure 3.7: Aggregation of one dimensional time series. For each of the different aggregation intervals ranging from 5 minutes to 1 day, we show all points in time where our z-Score based event detection algorithm would have reported an event. The time slice which contains the "gold standard" anomaly (defined by Numenta Anomaly Benchmark [111]) is thereby highlighted in green. Note that for aggregations 12 hours and 1 day, the peek frequency for this target anomaly disappeared almost completely. The aggregation of adjacent frequencies now correspond to the same time frame and thus smooth out small peaks.

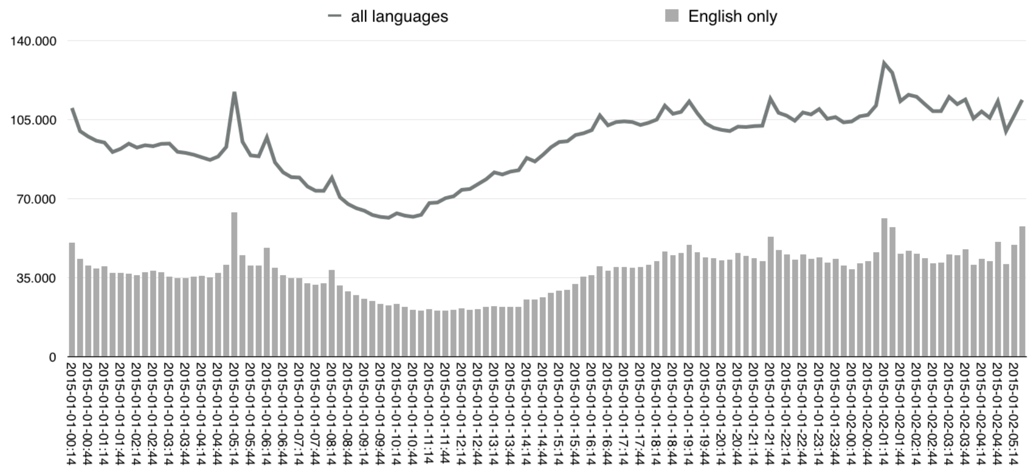


Figure 3.8: Twitter usage varies due to day-night cycles of twitter users. Absolute popularity is subject to seasonal trends such as working hours and weekdays. Normalization is thus needed to not “detect” anomalies that only reflect those seasonal patterns.

Chapter 4

Incorporated Publications and Co-authorship

Publications included in this thesis have been developed mainly with researchers from the Database Systems Group of the Ludwig-Maximilians-Universität München (LMU). The doctoral adviser Prof. Dr. Hans-Peter Kriegel supervised all of these publications. Publications discussed in Chapters 5, 6 and 9) have been a joint work with Dr. Erich Schubert. Additional co-authors include PD Dr. Arthur Zimek (Chapter 9) and Dr. Andreas Züfle and Dr. Tobias Emrich (Chapter 13). Due to this co-authorship, the contribution of this thesis' author is outlined in this chapter. Additional publications of the author that are not in this thesis are *Robust Segmentation of Relevant Regions in Low Depth of Field Images* [73], *Robust Image Segmentation in Low Depth Of Field Images* [74] and *Video route* [67].

4.1 SigniTrend: Scalable Detection of Emerging Topics in Textual Streams by Hashed Significance Thresholds

This publication [163] was a cooperation between Erich Schubert, Michael Weiler and Hans-Peter Kriegel. The author of this thesis contributed the initial idea to use probabilistic counting with multiple hash functions and a specific bucket update strategy that minimizes the approximation error. This idea was implemented by this thesis' author as an initial proof-of-concept prototype that uses probabilistic counting similar to Count-min sketches [58]. The other authors contributed an optimized implementation, design and experienced writing and provided guidance throughout the development process. The author of this thesis was also responsible for the data and majority of the experiments (including additional experiments that did not make it into the final publication) and presented this work at the 20th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), New York in 2014.

4.2 SpotHot: Scalable Detection of Geo-spatial Events in Large Textual Streams

This publication [166] was a cooperation between Erich Schubert, Michael Weiler and Hans-Peter Kriegel. The author of this thesis evaluated different ideas and strategies for integrating geographic information into SigniTrend: the idea to map coordinates to their textual hierarchical administrative boundary counterpart was developed as a proof of concept by this thesis' author. The other authors contributed with an enhanced high-resolution geographic lookup, integration into the previous codebase, and provided feedback, guidance, and experience with the publication process.

The author of this thesis is also responsible for the evaluation of the method by implementing comparison methods and joining the results with other data sources for comparison. The author of this thesis contributed with large parts of the related work, and the majority of the experiments, such as the runtime and scalability experiments, for which the author of this thesis implemented several related algorithms, particularly GeoScope. Further, experiments with detecting earthquakes and the therefore necessary data preparation was done by this thesis' author as well as the implementation and clustering for the experiment on the WikiTimes data set [184]. This work was presented by the author of this thesis in 2016 at the 28th International Conference on Scientific and Statistical Database Management (SSDBM) in Budapest (Hungary).

4.3 Scalable Detection of Emerging Topics and Geo-spatial Events in Large Textual Streams

This publication [165] was submitted as a short paper summary of Signi-Trend [163] and SpotHot [166] at the LWDA 2016 Conference in Potsdam, Germany. The authors and their personal contributions of this publication remain the same as from the individual publications.

4.4 Outlier Detection and Trend Detection: Two Sides of the Same Coin

This publication [167] was a cooperation between Erich Schubert, Michael Weiler and Arthur Zimek. The author of this thesis mainly provided the input regarding Event Detection and the motivation for the relationship of events and outliers. Outlier Detection was provided in particular by Erich Schubert and Arthur Zimek. This publication was presented by the author of this thesis at the 1st International Workshop on Event Analytics using Social Media Data (EASM) in 2015, Atlantic City, United States of America, in conjunction with the IEEE International Conference on Data Mining series (ICDM).

4.5 Geo-social Co-location Mining

This publication [193] was a cooperation between Michael Weiler, Klaus Arthur Schmid, Matthias Renz and Nikos Mamoulis. The author of this thesis contributed large parts of the related work and the conceptual idea of this publication as well as the implementation of the grid based spatial index for efficient nearest neighbour queries. Additional contributions include the implementation of the occurrence probability estimation and the calculations of the possible world semantics with generating functions.

4.6 Socio Textual Mapping

This publication [194] was a cooperation between Michael Weiler, Andreas Züfle, Felix Borutta and Tobias Emrich. The author of this thesis provided the conceptual idea of this publication as well as the motivation. The complete implementation was provided by the author of this thesis as well as experimental results and the clustering and term scoring formulas. Additional contributions include the formalisation of the scoring functions based on the occurrence probabilities.

4.7 TrendTracker: Modelling the Motion of Trends in Space and Time

This publication [161] was a cooperation between Klaus Arthur Schmid, Christian Frey, Fengchao Peng, Michael Weiler, Andreas Züfle, Lei Chen and Matthias Renz. The author of this thesis contributed large parts of the conceptual idea and provided methods from the event detection algorithm publications discussed above as well as related work regarding event detection.

Part II

Trend Mining and Event Detection

Chapter 5

Emerging Topic Detection

5.1 Introduction

Traditional text mining techniques such as clustering and topic modelling cannot be used trivially when processing a live stream of data. They can provide meaningful insight to refine e.g. a search query or to analyze a static text corpus, but are not applicable to fast-flowing, ever changing data streams. To cope with this flow of data, emerging topic detection is a useful tool, yet itself an emerging area. The goal of emerging topic detection is to identify new events early, to notify the user of the evolving story. Intuitively the goal is to have an automated "breaking news" detection. If the user is able to customize and prioritize the data sources, this will ultimately allow personalized breaking news and event detection. Corporations could use the additional knowledge to quickly react to future market needs and thus increase their profit. But the detection of trends in such fast-flowing data remains challenging because most documents contain raw unstructured natural language text; often abbreviated such that the interpretation is difficult for both, humans and computers. Ambiguity, homonyms and synonyms further add to these challenges. Even if some semantic annotations like tags are available, these are not necessarily helpful for the problem at hand, as they may be ambiguous as well and not all of them have an actual meaning [108]. Due to these problems, the resulting analysis quality still does often not meet the expectations in practise.

Scalability of textual analysis continues to be a problem. Many approaches are based on simple preprocessing such as stop word removal and stemming, and then use distributed algorithms to simply count and track word occurrences over time. Very few methods seem to be able to extract meaning from sentences, correlate words and model complex information in near real-time on high volume data [49]. This may explain why text analytics seems to take off very slowly.[1]

In this chapter, we propose a statistics-based score for evaluating trends as

[1] `http://www.kdnuggets.com/2014/02/poll-results-text-analytics-use-shows-no-significant-change.html`

well as scalability improvements to allow tracking arbitrary word pairs. The detected trends are then aggregated into a cluster to be further analyzed by the user.

5.2 Challenges

The analysis of social media data poses several challenges: first of all, the data sets are very large, secondly they change constantly and third they are heterogeneous, consisting of text, images, geographic locations and social connections.

Social media such as Twitter produces a fast-flowing data stream with thousands of new documents every minute (see Section 3.10). Many traditional analysis methods do not work well on such data: the informal, often abbreviated language with many special technical terms, emoji icons and constantly changing portmanteau words prevents many advanced linguistic techniques from understanding the contents. Furthermore, many advanced methods in topic modelling need to visit data multiple times to optimize the inferred structure, which is infeasible when thousands of new documents arrive every minute. Thus, algorithms for such data sets are usually designed around simple counting, or require the data set to be reduced via predefined keywords and location filters. Such algorithms based on simple counting of the most frequent words and phrases often yield unsatisfactory results because the output is dominated too much by quantity: the most frequent terms usually contain only well-known information such as the popularity of Justin Bieber on Twitter, "good morning" in the morning, "dinner" at night and the TV series currently on air. As a result, more interesting events are then hard to discover due to the quantity of everyday chat. Using absolute counting for event detection only works if the events have already been reported on TV and other mass media and we observe the global echo in social media. Instead, we need approaches that can identify *significant* patterns without reporting the obvious over and over. Due to spam and noise, not every single text is interesting, but identifying the interesting subset is hard.

5.3 Related Work

The field of emerging topics detection receives much attention, as the daily flood of information available in social media and news agencies is impossible to overlook – in fact, the media are reporting on the increasing volume of data every day. To know what topics are currently "hot" could be an enormous advantage both for private uses and businesses. To solve these demands, commercial systems like Dataminr, Sysomos, Brandwatch, MediaMiser and Topsy have been established. There are also a number of non-commercial systems like the CMU system [201], the UMass system [20], Meme-Detection System [116] or Blogscope [31]. Most of the published research focuses on Twitter as primary source of information inter alia due to its free public streaming API[2] for accessing random samples of all public statuses and the contained hashtags, which help to reduce the dimensionality of the incoming text and could provide some additional semantic information.

The UMass system [20] represents documents in a vector space such as a term frequency vector. Incoming documents are first compared to the database of previous documents using a nearest-neighbor search and then are added to the corpus. Documents that differ enough from the most similar earlier document are considered to be first stories. This approach, however, does not scale well to large data sets such as Twitter due to the need to maintain the database corpus [146]. To scale this nearest-neighbor approach to Twitter streams, the use of locality sensitive hashing (LSH [84]) was proposed [146]. To limit database size, buckets are limited in size and the oldest document is removed from the bucket when the bucket would overflow.

Kleinberg [93] uses an infinite state automaton to model frequency bursts of incoming documents such as emails. Different states of the automaton correspond to different message frequencies, and a hierarchy is extracted from the state transitions. In Blogosphere [148], Kleinberg's approach was applied to the titles of blog posts. To improve performance, they had to omit keywords co-occurring with other keywords in the same title. In a post-processing stage they enrich bursty keywords with semantically correlated terms by looking at the five nearest neighbors using an Euclidean-based distance metric on the automatons' state series. Kleinberg's burst modelling was also used by Takahashi et al. [178], where it was applied to topics estimated by a dynamic topic model (DTM).

Cataldi et al. [46] proposed a method that extracts terms from tweets to model a life cycle inspired by biological processes. They define a term as *emerging*, if it is now frequent but was rare in the past. To improve results, they also consider social relationships (like the followers count) to determine the user authority. Although they do a post-processing step to enrich detected trending terms with semantically related keywords, they cannot detect when two keywords that have already been frequent before are frequently co-mentioned such as when *Obama* and *Merkel* meet.

TwitterMonitor [132] first identifies bursty keywords that have a higher absolute frequency than usual and uses them as a seed to explore them further. Keywords that co-occur in the same tweets (within a small history window) are grouped together. For each such keyword set a singular value decomposition (SVD, [63]) is applied on all tweets containing them. The extracted terms are used to build a better description for each bursty keyword set.

EDCoW [196] applies a wavelet analysis on words to model their frequencies. Trivial words are identified with their cross correlation value. In Hip and Trendy [141], the focus lies on building a taxonomy from trends for a specific geographic area and to categorise them, rather than finding co-occurrence

[2]`https://dev.twitter.com/docs/streaming-apis`

trends. Similarly, CiteSpace II [51] has a strong focus on visualizing emerging trends.

Density estimation on data streams as performed in stream clustering approaches such as CluStream [12] is also related to our approach. However these algorithms assume you have a stream of coordinates, and want to estimate the resulting coordinate-based density. Such vector data arises for example in sensor networks, where the need of compressing historical information [64] can be satisfied using wavelets, discrete cosine or histograms. Temporal velocity profiles [10] are then applicable to such data and can be used to predict the positions of dense regions of moving objects. In our approach, we will be estimating a density on the temporal domain only, as if we would split the input stream into separate streams for each word co-occurrence, and process them separately. These techniques therefore do not apply here.

The problem of top-k monitoring [30] is related, but it does not take relative significance into account. These approaches, often aimed at detecting denial-of-service (DOS) attacks in network flows, are interested in the most frequently occurring patterns only. In the context of monitoring textual streams, these approaches would only monitor popular terms, that are most of the time uninteresting.

Exponential histograms [62] have been proposed for probabilistic counting as well as for weakly additive functions. This was then extended to continuously monitor variance over data streams [29]. As our method also relies on variance, this work is closely related. However, we use a simpler yet effective approach: their techniques are designed with the requirement to take exactly the previous N observations into account. Our approach uses exponential weighting, which will never fully forget data, but aggregates historical values into a single figure, in which old results will eventually have a weight of effectively zero.

A rather similar approach to our system is enBlogue [21]: Like our approach, they are also interested in the co-occurrence of words, instead of relying on single terms. The central measure of their approach is Jaccard similarity [85] of two words in relation to their overall popularity. Like most other approaches processing Twitter data, they first preprocess the data, in order to extract hashtags and named entities as tokens. In order to find co-occurrences, they start with a reduced seed list of frequent tokens; then track all token combinations that contain at least one of these seeds. For each token pair, a short history of ρ recent occurrence counts and popularities is kept; token pairs not seen during the last ρ epochs are discarded. Similar to our method, exponential smoothing is applied to predict future values from history. A key limitation of enBlogue is the need for controlling the memory usage. A number of mechanisms are employed to reduce the data volume: only hashtags and named entities are used for the analysis. Then seed tags are chosen based on a minimum occurrence threshold as well as a top-k filter. However, the experiments indicate that the accuracy drops linearly with the seed tags parameter, so these thresholds do not appear to be beneficial. Since enBlogue keeps a history of 2ρ historical values for any pair of tags tracked, it scales badly with respect to memory usage.

In our experiments, we observed as many as 700 million word and word pairs (see Table 7.2). While not all of them will be kept in memory at the same time, this requires a substantial amount of memory and update cost.

The method we propose improves over enBlogue in multiple ways. Instead of keeping a history, we only need two floating point values per record. Secondly, by storing this data only approximately in a hash table, we are not as much affected by the large number of potential word pairs. As we exploit hash collisions, we only need the hash table to be large enough to store all frequent word pairs, which due to the long-tailedness of the distribution is a substantially smaller number (as seen from the quantiles in Table 7.3). Petrović et al. [146] proposed a method for first story detection based on locality sensitive hashing. While our approach borrows on ideas from LSH and MinHash, we do not use this for nearest neighbor search, and do not need to store the individual documents for similarity search.

5.4 Detection of Significant Topics

In information retrieval and text mining, a popular similarity measure for text is cosine similarity on the term frequency (tf) vectors, weighted by the inverse document frequency (idf). This model is commonly referred to as tf-idf vector model (see Section 3.5 for a brief introduction). Depending on the definitions used, term frequency can either be the absolute counts of each term, or the relative frequency. In the context of trend detection, such similarity measures are difficult to use: they are designed to quantify the similarity of two documents, but trending topics may span many documents, and documents will often cover more than one topic. In particular, topics may have subtopics; and within a larger topic (such as pop music) a subtopic (such as a particular artist) may be trending, even when the larger topic is not.

A naive approach to perform topic detection would be the use of cluster analysis. However, cluster analysis on streaming data is far from a solved problem. Often, these clustering algorithms are unable to recognize hierarchies of clusters, and may need parameter fine tuning. Because of these limitations, the algorithms are mainly useful to cluster the result set of an information retrieval task for user presentation. Another similarly naive approach maintains a database of recent documents, and searches the nearest neighbor (i.e. the most similar earlier document) for each new document. As shown in [151], nearest-neighbor distances are a popular measure of outlierness. While this works reasonably well in a controlled corpus, it will fail on many noisy real data sets. While a low nearest-neighbor distance is indicative that the document is a near duplicate, the contrary does not hold: not every document will be the start of an interesting emerging topic, but it may also be just noise.

When detecting emerging topics in data streams, we cannot assume that we already have a cluster for the topic. In fact, it is desired to detect the topic as soon as possible. Yet at the same time, we are only interested in topics that both have a minimum size, but also show an "unusual" growth rate. When looking for trending subtopics, this becomes even more challenging – here, we may not be interested in the main topic, but only in the restriction to a subdomain.

In the following, we will discuss some important aspects for emerging trend detection. First of all, we discuss how to measure *significance* of trends, as opposed to just some deviation score. Secondly, we will elaborate briefly on the *relationship to Outlier Detection*. Then we will discuss the importance of *word co-occurrences* for detecting trends that may be masked otherwise and additional challenges for *early detection* when using the suggested statistics. Finally, we will discuss *hashing* approaches for *scalability* to monitoring all pairwise co-occurrences.

5.4.1 Emerging and Trending Topics

When users see "trending topics", they usually assume that these are simply the most popular terms. However, users do not only want popularity, but they also expect *novelty*. Therefore, we must not simply look at the most popular tags, but we must take this popularity into its historical context. As noted in Section 3.11.2 we normalize observed frequencies by dividing its absolute frequency by the total number of documents for the day. This avoids "seasonal" patterns that arise - depending on the data source - at working hours or weekdays. Using these normalized frequencies proved much more robust in our experiments.

To evaluate the significance of a trend, we suggest to make use of statistical best practice such as the z-score (formally, the z-score assumes a normal distribution; while this will likely not hold, it nevertheless can serve as a reasonable heuristic for our purposes):

$$z(x) := (x - \mu)/\sigma \tag{5.1}$$

Where x is the frequency of a currently observed term, μ its corresponding expected mean frequency and σ its standard deviation. To use this on data streams, we need a moving average and moving standard deviation such as the exponentially weighted average ($EWMA$) and the associated standard deviation or variance ($EWMVar$):

$$sig(x) := \frac{x - EWMA}{\sqrt{EWMVar}} \tag{5.2}$$

in order to increase stability (the variance could be 0) as well as to account for non-interesting fluctuations of rare terms, we will ensure a minimum average and minimum standard deviation of β, which we call the bias term:

$$sig_\beta(x) := \frac{x - \max\{EWMA, \beta\}}{\sqrt{EWMVar} + \beta} \tag{5.3}$$

This bias term β not only avoids a division by 0, but also serves as a noise filter. For low volume data $EWMA < \beta$ with a small value of β we cannot statistically argue about trends. Intuitively, we should set β to the expected background noise level, i.e. how often rare terms occur naturally in the data stream without trending. As previously shown in Section 3.4 typical term frequencies are long tail distributed, which means that we will encounter many terms that should be handled as noise. The bias term also takes into account that our input data is discrete – there are no "half occurrences" of words.

In order to estimate the average *EWMA* and the variance *EWMVar* for a data stream and a learning rate α, we can rely on earlier work by Welford [195] and West [197] on incremental mean and variance. The update equations given by Finch [68] for the exponentially weighted variants allow these values to be efficiently maintained on a data stream:

$$\Delta \leftarrow x - \mathit{EWMA}$$

$$\mathit{EWMA} \leftarrow \mathit{EWMA} + \alpha \cdot \Delta \tag{5.4}$$

$$\mathit{EWMVar} \leftarrow (1-\alpha) \cdot (\mathit{EWMVar} + \alpha \cdot \Delta^2) \tag{5.5}$$

The learning rate α can be set using the half-life time $t_{1/2}$; the time when the weight of data has reduced to half of the initial weight. With this parameter a domain expert will be able to choose easily based on his experience and needs:

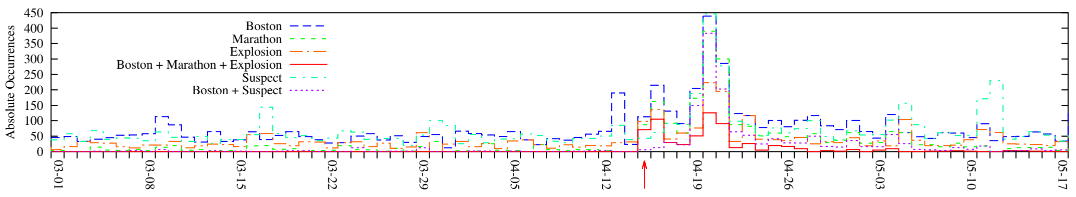
$$\alpha_{\textit{half-life}} = 1 - \exp\left(\log\left(\tfrac{1}{2}\right)/t_{1/2}\right) \tag{5.6}$$

This update equation is best used with a fixed update cycle. While we could adjust the update cycle dynamically by adjusting α or $t_{1/2}$ accordingly, there are good reasons not to update too often: first of all, a fixed recomputation interval gives better performance guarantees, and secondly we have more control over the statistical validity of our estimates. We cannot use above equations to update the statistics on every arriving record, but we must perform some aggregation to estimate the popularity x reliably. When using a too high update rate, we will have more variance in our estimation of x, which will in turn harm the estimation of *EWMA* and *EWMVar*. When using a data source with a natural cycle, such as a news ticker having a daily pattern, we can expect best results aligning our updates with this cycle.

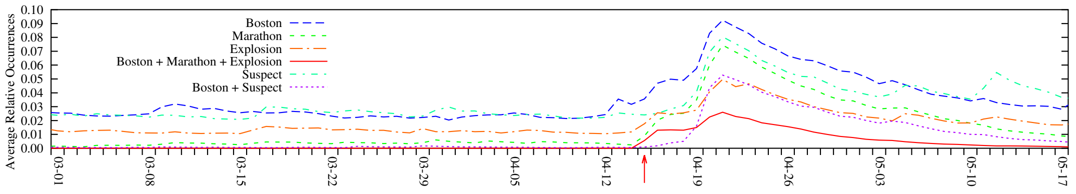
Note that we avoid the popular equation $E(X^2) - E(X)^2$ for estimating the variance, because this equation is prone to numerical instability due to catastrophic cancellation with floating point arithmetic. This instability may be the reason why variance on data streams seems to be rarely used yet. Equation 5.5 is numerically stable [68], and thus preferable.

5.4.2 Emerging Topics are Outliers

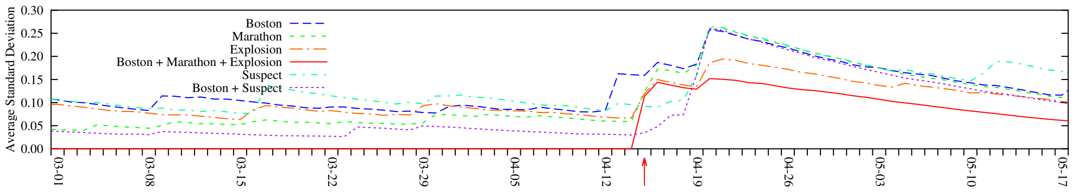
Emerging topics, when defined as "a set of documents which grows faster than expected from comparable other document sets", essentially are a specific kind



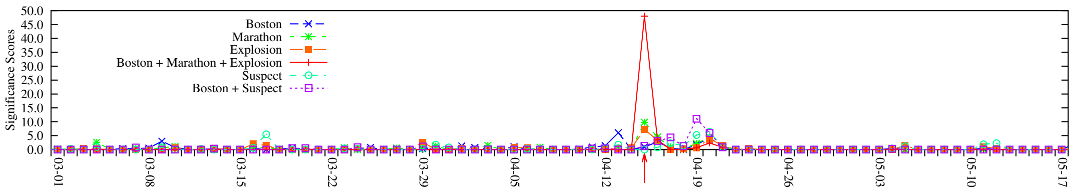
(a) Boston Marathon event – raw absolute occurrences



(b) Boston Marathon event – *EWMA* averages



(c) Boston Marathon event – standard deviations ($\sqrt{EWMVar}$)



(d) Boston Marathon event – significance scores

Figure 5.1: Visualization of selected word occurrences for the Boston Marathon bombing on news data

of outliers. In Chapter 9 the relationship of trends and traditional Outlier Detection methods is discussed in more detail.

A similar interpretation for Outlier Detection is possible for our approach: for each topic (approximated as word or word pair), we compute a reference model consisting of an average frequency and variance based on an exponentially weighted *temporal neighborhood* set. We then compare the current frequency to this reference, and measure the trend by the deviation from this model.

The relationship to Outlier Detection not only exists on a formal level, but we are also seeing many problems typical to this domain [210]: we do not know beforehand what data we are searching for, and we do not have labeled training data so that popular machine learning approaches such as random forests cannot be trained for this problem. Instead, we have to fight the problems of masking and swamping [32, 77], where for example a continuously popular word such as *obama* may prevent the detection of related trends, or a single strong trend may mask other trends in the data set. In Chapter 9 we will discuss the relationship of Event Detection and Outlier Detection in more detail.

5.4.3 Trend Detection on Co-occurrences

Equation 5.4 and Equation 5.5 can be applied to any numerical variable x. In order to apply this approach to trend detection in textual data, we need to represent the incoming data stream appropriately. We will not only build one variable for each word, but also for each word co-occurrence. The reason is that we may be unable to reliably detect or analyze some trends on single words only. In Figure 5.1 we visualize the Boston Marathon bombing on news data. In Figure 5.1a the raw word occurrences are presented. The word *boston* shows a comparable peak three days before the Marathon. Figure 5.1b visualizes the EWMA moving average. In this figure, it becomes evident that *boston* is (unsurprisingly) frequently mentioned, but even *explosion* (as e.g. "the explosion of mobile traffic", "stock explosion" and "Dreamliner battery explosion") is occurring quite often. However, the exact combination of these three words first occurs on April 15 substantially. Using the proposed significance measure, as visualized in Figure 5.1d, shows a 48σ event for this combination. For other related words, this event of global media interest also shows a peak, but not quite as strong. On April 19, we observe a similar peak for the combination *boston*, *suspect*; on this day the suspects were identified and the manhunt began. This example demonstrates the benefits of tracking word co-occurrences instead of single words.

5.4.4 Early Detection of Trends

Equation 5.4 and Equation 5.5 are not designed for continuous updating of the *EWMA* and *EWMVar* estimates, but we first need to have a robust estimate of the value of x. While we can easily adjust $\alpha_{\textit{half-life}}$ to dynamic time windows, the value of x will become noisy and exhibit a too high variance to be useful for trend detection. A simple but reliable approach estimating x is to use temporal slicing on natural cycles and volume of the data source (which may be a day for a news feed, or an hour for Twitter). Even when not updating *EWMA* continuously, we can still perform an online detection of trends. In order to evaluate the significance of a trend, it is even desirable to compare the current estimate of x with estimations based on delayed data only, i.e. we want to compare the observed value x with the predicted value *EWMA* based on the previous day, and not including the latest data in the prediction yet. By solving Equation 5.2 for x, we can obtain an alerting threshold τ if we fix a desired significance niveau s:

$$x > EWMA + s \cdot EWMVar =: \tau \tag{5.7}$$

Intuitively, when $x > \tau$, we are observing an $s \cdot \sigma$ significantly increasing trend in the data. We do however need to acknowledge that this is not based on proper statistical hypothesis testing, so we cannot assume that 3σ events are rare: the 66-95-99.7% rule (the three sigma rule) does not apply here, as we will be monitoring thousands of trends in parallel and have some margin of error. As introduced in Section 3.9 of Part I, this is often called "multiple testing problem" [155].

On the other hand we are interested in seeing multiple events every day, and not just the most significant events of the year. This approach can be seen as a variant of Bollinger Bands as used in stock market analysis, except that our data sources do not exhibit the fast negative feedback loop driving the stock market.[3]

The main difficulty with this approach is getting a robust estimate of x while the epoch is not yet complete. On the news data set, for example, it is common not to see any news posted before 6am in the morning. The first news item each day will naturally produce terms with a naive document frequency of 1; yet they do not constitute a reliable trend so far. However, we also do not know how many news will be posted during the day in total. Therefore, in order to estimate x, we need to learn to predict the number of documents to be posted during the epoch; we can then compare the absolute number of occurrences, the number of documents seen so far, and the number of documents expected to obtain a better estimate of x.

5.5 Scalability by Hashing

Although Equation 5.4 and Equation 5.5 are efficient to compute, we would still need to perform this for any word pair on our data stream. Computing the statistics for every pair of tokens would quickly exhaust memory resources and is as seen in Table 7.2, not scalable when using naive methods. Instead of restricting the set of candidates to monitor beforehand, we propose to use a probabilistic approach based on hash tables.

The popularity of words follows a long-tailed distribution: the majority of words have a low occurrence rate in the data. This property can be exploited using hashing techniques. Our approach is related to heavy-hitters algorithms such as Bloom filter [38], MinHash [40] and Count-min sketches. As introduced in Section 3.3, Count-min sketches are data structures that represent a lossy and thus usually smaller approximation of the data. We make use of a modified Count-min sketch. The usage of these sketches allows us to speed up computations so that we can track the statistics of the large amount of words and word pairs. The sketches used in this work are lossy in the same way that a count-min sketch can overestimate the count of an object: the majority of word pairs in Twitter are unique combinations that will never constitute a significant event. By the update procedure discussed below, *the more frequent word pair wins* – this way, we are less likely to make errors on frequent terms, and we do overestimate the frequency of rare combinations. But since rare

[3]It is therefore unlikely that the proposed method is beneficial for stock market analysis.

combinations cannot be statistically significant, this error does not affect the output of the algorithm much. Because of such low-frequency collisions, we tend to overestimate the variance slightly, and thus report lower significances than we would obtain by expensive exact counting of all pairs. Sketches could also be used to generate candidates for exact analysis. However, the quality of the data obtained using our approach was so close to the exact results, that there was no need to further pursue this research direction.

As this data structure is only an approximation of the exact frequency we still need to design it in a way to reduce overall error. Our modification to achieve this goal is described as follows.

Each sketch consists of a table of size 2^b, where $b \geq 20$ produced excellent results as shown in Section 8.3. The hash functions $H_1(x) \dots H_h(x)$ used throughout the pipeline must be the same so that buckets are aligned with each other. The number of hash functions h is a small integer, and the value $h = 3$ was experimentally shown to be large enough and used throughout our experiments.
The following sketches and update procedures are used:

1. Simplified count-min [58] sketch (using a single table, instead of a separate table for each hash function). To update, read buckets $H_1(x) \dots H_h(x)$, compute the minimum, then increment buckets $H_1(x) \dots H_h(x)$ only if their current value was the minimum.

2. Count-min sketches that additionally store the last seen word or word pair of the bucket. If we increment the counter in the count-min sketch, we also update the word pair attached to the bucket.

3. SigniTrend sketch, which stores the exponentially weighted moving average and variance (see Equation 5.5 for details).

4. Threshold sketch, which stores the standard deviation and last reported significance. At the end of each epoch, the last reported significance is heuristically reduced by multiplication with 0.75, to allow events to recur after a pause. The standard deviation is simply obtained by computing the square root of the variance stored in the SigniTrend sketch, to avoid repeated calculations.

Operations on the sketches are either confined to the affected hash buckets $H_1(x) \dots H_h(x)$, or affect all buckets the same way, and can be efficiently executed with SIMD[4] operations. Instead of probabilistic set membership testing, our hashing scheme is designed for providing an upper bound of the *EWMA* and *EWMVar* values. We use 2^ℓ buckets (each storing an *EWMA* and *EWMVar* value), and k hash functions, so that each word is mapped to at most k buckets. When updating the *EWMA* hash tables – when transitioning from one epoch to the next – we first hash each word (with a frequency larger than

[4]Single Instruction, Multiple Data

β) into its k buckets. For each bucket we track the *maximum* popularity x observed in the current epoch. Next, we update the *EWMA* and *EWMVar* values in the hash table with the *maximum* x observed for all words in this bucket only. When computing the alerting threshold τ for an observed word w, we again inspect the k buckets given by the hash functions. The threshold is then obtained by taking the *minimum* alerting threshold of all buckets (Equation 5.7). Assuming there exists at least one hash bucket where the candidate word w has the maximum popularity x_w of all words in this bucket, then the *EWMA* value stored in this bucket will not be overwritten by a different word w'; and at the same time none of the k buckets was last updated with a lower popularity than x_w. Otherwise, i.e. if in each of the k buckets there was a hash collision with a more popular word w', we will overestimate the *EWMA* and *EWMVar* values, and we may miss an emerging trend. To avoid this, we need to choose ℓ large enough, such that the majority of bins are not filled with frequent keywords.[5]

Using this hashing strategy, we can control the amount of memory and computation needed for maintaining the trend statistics very well. In fact, this allows us to scale our approach beyond tracking single word (or hashtag) occurrences to monitoring even word-pair occurrences in large data streams. Word-pair occurrences prove very effective in our experiments at detecting subtopics. For example when Edward Snowden traveled from Hong Kong to Moscow, all of the individual words {*edward*, *snowden*, *hong*, *kong*, *moscow*} themselves have not been trending (events surrounding Edward Snowden have continuously been in the media at that time), but various combinations of these words such as *snowden* and *moscow* exhibit a significant peak.

The use of hashing yields a number of benefits. First of all, updating the statistics becomes a vectorized bulk operation. Secondly, the memory usage is constant and the data can easily be serialized either for transmission to a different system, but also for checkpointing and crash recovery. After loading the last checkpoint, either a replay of the latest data can be used to restore the exact state, or the system can even decide to only process new data, and rely on the statistics to recover (it may then be desirable to disable alerting for this epoch).

5.5.1 Trend Redundancy and Refinement

A secondary challenge is the redundancy of trends. Assuming that a word such as *snowden* is trending, then we will likely see other related words such as *edward* also trend at the same time. But we may also see uninteresting combinations such as {*snowden*, *the*} trend significantly compared to previous usage of this word combination. Probably the most important step here is to perform stop word removal: by not including known stop words in the trend analysis, we both have to analyze much less pairs, but we will also remove a

[5]This statistic can easily be monitored; in our experiments we chose $\ell = 22$, which yields about 4 million bins but keeps the hash table size at a few megabytes of memory.

Algorithm 1: Document Processing

```
Data: epoch Epoch identifier
Data: $EWMA[c]$ Averages hash table
Data: $EWMVar[c]$ Variance hash table
Data: $h_i$ Hash functions
Input: $s$ Threshold for refinement / alerting

open index[epoch] shard for writing
initialize frequency map
initialize stats map
/* Index a new document */
foreach doc in current epoch do
    foreach unique word and word-pair in doc do
        /* Update refinement index */
        add doc.id to index[epoch][word]
        /* Update current word counts */
        increment frequency[word]
        /* Get word statistics from hash table */
        if not stats[word] then
            $(\mu, \sigma) \leftarrow (\infty, \infty)$
            foreach hash function $h_i$ do
                $c \leftarrow h_i(word)$
                if $EWMA[c] < \mu$ then
                    $\mu \leftarrow EWMA[c]$
                    $\sigma \leftarrow \sqrt{EWMVar[c]}$
                end
            end
            $stats[word] \leftarrow (\mu, \sigma)$
        end
        $(\mu, \sigma) \leftarrow stats[word]$
        /* Test for significance threshold */
        $x \leftarrow$ estimate frequency of word
        if $(x - \max(\beta, \mu))/(\sigma + \beta) > s$ then
            send to refinement for early alerting
        end
    end
end
```

Algorithm 2: Perform end-of-day analysis (continuation of Algorithm 1)

```
Data: epoch Epoch identifier
Data: $EWMA[c]$ Averages hash table
Data: $EWMVar[c]$ Variance hash table
Data: $h_i$ Hash functions
Input: $s$ Threshold for refinement / alerting

/* Perform end-of-day analysis                                  */
initialize trending topics list
foreach unique word and word-pair in frequency do
    $(\mu, \sigma) \leftarrow stats[word]$
    /* Test for significance threshold                          */
    $x \leftarrow frequency[word]/|documents|$
    if $(x - \max(\beta, \mu))/(\sigma + \beta) > s$ then
        add word to trending topics
    end
end

close index[epoch] shard for writing
Refine trending topics
Cluster trending topics
Produce end-of-day report

/* Update the statistics table for next epoch                   */
/* Aggregate into maximum for each bucket                       */
initialize update-table
foreach word and word-pair in frequency do
    frequency $\leftarrow$ frequency of word
    foreach hash function $h_i$ do
        $c \leftarrow h_i(word)$
        if frequency > update-table[c] then
            update-table[c] $\leftarrow$ frequency
        end
    end
end

/* Update statistics table                                      */
foreach hash code c do
    freq $\leftarrow$ update-table[c]/number of documents
    $\Delta \leftarrow freq - EWMA[c]$
    $EWMA[c] \leftarrow EWMA[c] + \alpha \cdot \Delta$
    $EWMVar[c] \leftarrow (1 - \alpha) \cdot (EWMVar[c] + \alpha \cdot \Delta^2)$
end
```

large share of uninteresting co-trends. In addition to a language-specific set of stop words, it is also beneficial to include domain specific stop words, such as *follow*, *tweet*, *retweet* (*rt*), *lol* and *twitter* for Twitter.

In the refinement phase, we can use an inverted index to both verify observed trends (as the hash table may have had a collision), but also compute the overlap between any two words involved in the detected trends. For this we employed the clustering toolkit ELKI [8], and used hierarchical clustering with Ward linkage. Similarity is measured by how significant the two words trend together. We cannot assume that related words form a clique: consider for example *fiscal*, *cliff*, *barack* and *obama*. The last two will usually not qualify as a trending topic, but the names are in fact one of the most mentioned words in news on any day; however all four together form a known topic. In Algorithm 1 and Algorithm 2 we give a pseudocode for the overall detection process.

Alternatives at this stage – which we plan to investigate in future work – include finding maximum-weight cliques (as used in [109]) and topic modelling techniques such as pLSI and LDA. However, topic modelling is not guaranteed to produce a more meaningful output either [49].

Chapter 6

Geo-spatial Event Detection

In this chapter, we focus on detecting events consisting of text and location information, and introduce an analysis method that is scalable both with respect to volume and velocity. We also address the problems arising from differences in adoption of social media across cultures, languages, and countries and incorporate novelty in our event detection by efficient normalization.

We introduce an algorithm capable of processing vast amounts of data using a scalable online approach based on the Event Detection algorithm discussed in Chapter 5, which is able to identify unusual geo-textual patterns in the data stream without requiring the user to specify any constraints in advance, such as hashtags to track. In contrast to earlier work, we are able to monitor every word at every location with just a fixed amount of memory, compare the values to statistics from earlier data and immediately report significant deviations with minimal delay. Thus, this algorithm is capable of reporting "Breaking News" in real-time as they happen in social media. This is achieved with an architecture that – as discussed in Section 5.5 – is optimized for data throughput with an efficient hashing strategy that tracks only significant parts of the data.

Location is modeled using unsupervised geometric discretization and supervised administrative hierarchies, which permits detecting events at city, regional, and global levels at the same time. The usefulness of the approach is demonstrated using several real-world example use cases using Twitter data. We show reportings of significant earthquakes and relate observed events to Wikipedia articles.

6.1 Motivation

As discussed in Section 5, we can efficiently detect unexpected frequent events and thus be able to answer the following questions:

What is the event: This is represented by the tokens and their combinations within a detected cluster.

When the event occurred: First significant occurrence when we observed an unexpected large increase that exceeds our thresholds.

But it cannot answer *where* an event took place. Because "concerts" or "earthquakes" are always happening around the globe, we may miss some events, as they will not be significant enough. As we will later see in our experiments of Section 8.3, we can incorporate location data to be able to detect a locally significant concert or earthquake. Furthermore, when users are not evenly distributed such "mainstream" behavior tends to obscure interesting developments in other parts of the world.

When the attack on Charlie Hebdo occurred at around 11:30, it still took 45 minutes to yield a significant number of mentions of the hashtag *#CharlieHebdo*, all of them originating in Paris. By 13:00, the news then had spread to the UK and by 13:30 most European countries were discussing the attacks on social media. By 14:00, the new hashtag *#JeSuisCharlie* had emerged.

While these quite substantial delays even on a major incident, indicate that the promise of real-time detection of events on Twitter may be overly optimistic and that TV continues to be of critical importance to news distribution. These observations demonstrate how geographic locality may help determine the importance to understanding an event. It also demonstrates the usefulness of a tool capable of analyzing such developments with little delay, without having to neither rerun the analysis, nor specify the topics of interest in advance. While *Charlie Hebdo* yields an obvious portmanteau keyword to use as hashtag on Twitter, this is not always the case and we need to monitor words that have not been manually chosen as a keyword by the user.

Thus, we are interested in a system capable of tracking unusual occurrences of arbitrary words at arbitrary locations all over the world in real-time, without having to specify the terms of interest in advance. Absolute counts—and the absolute increase—need to be adjusted for differences in usage.

6.2 Requirements

An algorithm and system that tracks all possible words at all possible locations needs to be carefully designed for scalability because of the high volume and velocity of social media data sources.

Geographical Proximity: An important relevance factor is geographical proximity. People tend to have a stronger interest in things related to their own location; thus newspapers offer various sections ranging from local city level to global world wide news. Motivated by editorial created news sections we want to create a system that is capable of including geographic metadata into the event detection process.

Track all Locations: An important requirement to our system is that the user does not have to specify desired locations beforehand, as we want to be able to monitor every possible location with a reasonable fine grained resolution such as urban districts. Likewise we will not restrict the vocabulary of our monitored terms to hashtags or top-k-frequent terms: every observed term should be considered in the scoring process. To enable such a limitation free tracking we focus on high performance throughput at the very first time we read the data from our stream. As shown in Section 8.3 even at the scale of Twitter, the resulting terms and locations, that receive an unusual strong frequency burst are rare.

Real World Application: It should be possible to apply the algorithm in a client-server architecture where the server does the statistical tracking. Each client can then easily filter out information that he or she is interested. As we will discuss in Section 6.5, systems that limit the locations beforehand are thus only usable for a single client (or a group of clients within the same regions of interest). Thus, the remaining candidates can easily be filtered by users on the client side instead of discovering evolutionary theme patterns [134] or discovering repeating patterns in time series [136].

6.3 Challenges

Velocity and Volume of the Data: When processing a Twitter feed, the obvious challenges are the velocity and volume of the data. We use a different feed which both has a larger volume and a higher rate of tweets with coordinates (at the cost of having no retweets), so that we have to process over 5 million geo-tagged tweets per day, between 3000 and 5000 tweets per minute. Thus, we can process over 5 million of geotagged tweets per day. Not all of these are usable – our spam- and duplicate filters will reject roughly a third of all tweets. The average rate to process at the analysis stage then are about 2500 tweets per minute (sometimes exceeding 4000 tweets/minute).

Spam: Twitter contains various kinds of spam in large quantities. There are advertisements for products, services, and Twitter followers, and some spammers try to manipulate the trending topics. But there are also teenagers sending mass love messages to their idols, bots announcing trending topics and weather forecasts, and “follow sprees” where users mass-follow each other to raise their follow count to appear more popular. We found the analysis results to become much more interesting when ignoring tweets that either include “follow” or “trend”, and using a streaming near-duplicate filter to drop much of such spam. We utilize a basic streaming near-duplicate filter to drop most spam and thus remove many false-positives, but do not further focus on this challenge in this chapter.

Variation of Tweet Density: Twitter adoption varies heavily from country to country: Twitter is popular e.g. in the USA, Brazil, Argentina, Indonesia, and Turkey. On the other hand, users from China, India, and Germany (where apparently Twitter usage and location sharing raise privacy concerns) are underrepresented in this data set.[1] If we want to obtain meaningful results for all areas of the world (including e.g. Berlin), we need to design an algorithm that is capable of adapting to local variations in the tweet density. Methods based

[1]The distribution of countries and locations within our data set can be found in Section 7.3.1

solely on counting the most popular terms will be biased towards countries with many users and unable to detect events in less populated areas.

Streaming Analysis: The number of tweets is not the sole parameter affecting scalability. The complexity of the analysis also influences the scalability a lot. Performing pairwise comparisons or computing a distance matrix is impossible if the data is too large to visit multiple times. Designing an algorithm that processes every record only once and that has to perform under substantial memory and time constraints (a streaming or "on-line" algorithm) yields a different scalability challenge. Many methods require the specification of a set of candidate words to track, or only track hashtags to reduce the data size by filtering. However, usage of Twitter changes over time, and hashtags only emerge after an event has occurred. Events are even more interesting if they are significant before people have agreed on which hashtag to use.

At the same time, single words may not be meaningful enough to identify an event. Thus, we need to design a system capable of tracking all the words and word co-occurrences at the same time if we want to produce the best possible results.

6.4 Problem Definition

Given a high-throughput textual data stream (too large to be effectively stored and queried) with geographic metadata, the goal is to identify terms that receive an unusually strong increase in their frequency as well as the locations and time associated with this unusual increase. The resulting events should not include spurious events that do not exhibit above characteristics, so that they can be used as unsupervised input for a "breaking news" detection system. The user must not be required to specify a limited set of candidate terms or locations, but instead the system must be able to learn the relevant vocabulary and locations from the data.

Therefore statistical significance scores need to be maintained for every single term as well as term-location pair. Thereby heavy hitters (items with high frequency) must not be underestimated. The frequencies of all observed terms and pairs need to be reviewed periodically to determine a specific threshold that defines the point from which on a single additional mention causes the corresponding item to be threaded as an event.

For each observed single term $t_i \in T$ as well as term-location pair $(t_i, l_j) \in T \times L$ the number of mentions within a defined time window (say 1 hour) must be monitored.

Because extracted word tokens from our textual data is long-tail distributed (recall Zipf's law from Section 3.4) we can use such approximation without loosing to much accuracy (see saturation experiment in Section 8.2.4).

6.5 Related Work

In scientific literature one may find a broad variety of applications tailored to specific tasks like the detection of earthquake events [157, 65], and the prediction of trends [106, 6, 27]. As we previously discussed in Section 5.3, most related work on event and emerging topic detection does not use geographic metadata.

In this section we want to highlight those publications that focus on geographical and location-based data coming from textual streams. Twitter itself publishes trending topics for a manually curated list of about 500 locations, with a limited choice of cities in each country. The exact algorithm used is not published, but it involves a "velocity" and does not include topics, like *#justinbieber* which are always frequent [2]. We assume it is based on a relative increase in frequency at a predefined location, i.e. the geographical information is used as a filter to split the data, but not in the actual analysis. Sakaki et al. [157] monitor Twitter users as sensors to detect earthquake and typhoon events. Signal processing algorithms such as Kalman filtering and particle filtering are used to handle the noisiness and to estimate the location of the earthquake. However, this approach requires that the user specifies query terms to select the topic of interest (e.g. $Q = \{\mathit{earthquake}, \mathit{shaking}\}$).

Crowd behavior is used to detect geo-social events in [115]. They partition the world into smaller regions of interest (RoI). For each RoI, they monitored the number of tweets, number of users and crowd movement. When unusual features are measured, tweets belonging to the corresponding RoI are reported as an emerging geo-social event. For performance reasons they avoid splitting regions into suburban areas, as "suburban areas will consume considerable unnecessary monitoring costs". Furthermore, their approach does not track the individual statistics of single words (or even pairs) and thus may miss events that do not match their 6-hour granularity.

Points of interest (POIs) are analyzed in [123] to identify knowledgeable expert users for predefined topics and POIs. This approach assumes that topics and POIs are reliably identified beforehand. Another approach [187] uses machine learning (e.g. Naive Bayes) to detect geo-spatial events. To keep the overall system efficient, they reduce the monitored locations by pre-selecting "a set of tweets based on their geographical and temporal proximity".

The motivation of Cataldi et al. [46] is similar to ours. They "believe that the flow of information directly rises in the geographical origin of the event and expands its influence proportionally to its global importance". However, they only consider the temporal aspect of tweets from communities and do not explicitly aggregate the individual locations. In addition they cannot detect when two keywords that have already been frequent before now are frequently co-mentioned.

Kim et al. [92] use predefined topics such as weather, TV and sport to build

[2]Source: `https://support.twitter.com/entries/101125`

a state level correlation matrix. To determine "hot topics" only a very small set of 9 social topics were analyzed. The article is not very clear how these topics were determined "semi-automatically" by looking at a term's frequency ratio $r_t^w = (f_t^w - f_{t-1}^w)(f_t^w + f_{t-1}^w)^{-1}$, where f_t^w denotes the term frequency of w at time t. For each topic, they build an adjacency matrix using Pearson's correlation coefficient on the U.S. state level.

Wang et al. [191] identify local frequent keyword co-occurrence patterns, but do not determine emerging or trending ones. One further drawback is the use of the Z-curve that leads to boundary effects as discussed and compensated in Section 6.6.

For a spatially limited subset it can still be feasible to count the number of users for every word and every location. Such approaches were used in [3, 4], but these approaches do not scale to large data sets.

EvenTweet [4] first identifies keywords by exact counting and comparison to historical information, then for every keyword analyzes the spatial distribution. For this, it needs to store large amounts of data and cannot operate on a live stream.

Other systems like Jasmine [192] solely focus on detecting local events and thus would miss trends of global importance like the Ebola virus outbreak, Super Bowl, or the acquisition of WhatsApp by Facebook.

Our approach is motivated by GeoScope [41], which tries to solve a similar problem (detecting events in geo-tagged tweets), but our algorithm is built upon an efficient significance model (discussed in Chapter 5) to overcome the limitations of GeoScope. GeoScope also uses Count–min sketch data structures [58] for approximate counting, but they independently use such tables for identifying the most frequent locations and the most frequent hashtags (the most frequent words would be stop words, thus the need to constrain their approach to hashtags; which is not a problem for our significance-based approach). An event in GeoScope is required to both having an unusually frequent term at a single location, and an usually frequent single location for this term. While this works well for extreme events that are observed in a single location only. However, events of global scale like the Super Bowl, which are watched all over the world will thus not be recognized as event. When decreasing the thresholds enough to capture less frequent topics and locations, the number of reported results grows exponentially. In our approach, we evaluate significance instead of frequency, and thus do not observe such spurious combinations. In their system, events are detected within a time window that must be specified by the user either quantitatively (e.g. 10^6 pairs) or temporally (e.g. 1 hour). Within a user-specified time window all frequencies of all observed location-topic pairs are tracked. They consider a location-topic pair (l_i, t_x) as "significantly correlated" if at least the fraction ϕ of all mentions from this location l_i are about the same topic t_x, and if conversely at least the fraction ψ of all mentions about topic t_x are from location l_i. As a result of this definition, topics that are discussed in many locations around the world within the same time window will not be considered events. Hence, events of global importance will neither be detected nor reported to the user. We will discuss this effect later in Section 8.3 when we evaluate our results.

When implementing their approach, limitations of GeoScope were revealed: Firstly, they approximate coordinates from geo-tagged tweets solely at a city level. Similar to the Z-curve approach from Wang et al. [191] this causes boundary effects when events are not tied precisely to a single city, but spread over multiple cities or just outside of a city (e.g. at a stadium or airport). In Section 6.6 we discuss in detail how our system handles such conditions.

Secondly, they monitor only cities that are at least "θ-frequent". Because they use $\theta = 0.005$ in their experiments, the maximum number of tracked cities is $\lceil 1/\theta \rceil = 1/0.005 = 200$. For each of these up to 200 frequent cities, they track the top "ψ-frequent" terms (up to $\lceil 1/\psi \rceil$ per city; $\psi = 0.05$, thus up to 20 topics per city). Similarly, they limit the number of topics tracked by $\lceil 1/\phi \rceil$ which with the suggested parameter $\phi = 0.05$ limits the tracking to 20 topics. Thirdly, they only consider hashtags as topic candidates; otherwise the top list would be filled completely with domain-specific stop words such as "follow" and "tweet". Our approach does not have this restriction, and does not suffer from stop words in the data much (it is more efficient to remove known stop words as early as possible though). For maintenance, they need to remove expired tweets from their counters, and thus need to keep a buffer of all tweets within the desired window size, which means additional memory cost. Again, our approach does not need this, as exponential aging handles the expiry of old data more efficiently.

6.6 Symbolic Representation of Location

We employ two different methods at the same time to generate a symbolic representation of the tweet location: a Grid-based token generation (Section 6.6.1) and an administrative boundaries token generation (Section 6.6.2).

6.6.1 Grid-based Token Generation

The first token generator is based on coordinate grids spaced at every even degree of latitude and longitude, which is an empirically chosen tradeoff between precision (the grid must not be too coarse) and abstraction: a too fine resolution reduces the chance of seeing a statistically significant number of mentions in the same location, while a too coarse resolution results in too many false positives. In order to avoid boundary effects (where the grid boundary would split a city into two separate grid cells), we use three overlapping grids, offset by one third and two thirds of the grid width. By the pigeonhole principle—as proven in [47]—we can guarantee that for every point there exists at least one grid, where it is not close to the cell border. Unless too close to the poles, we can thus guarantee that other points within a radius of about 20 miles are in the same cell, while events farther than 200 miles are in different cells. In-between

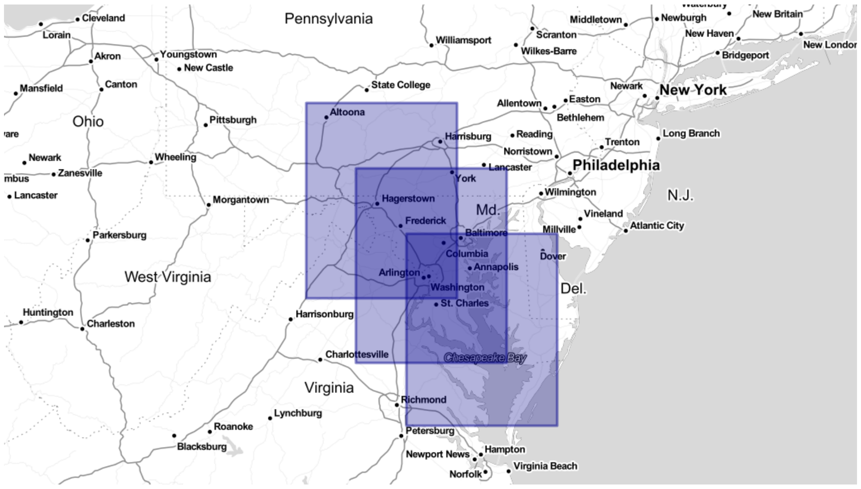


Figure 6.1: Grid tokens for a location in Washington, DC
Background © OpenStreetMap contributors, tiles by Stamen Design.

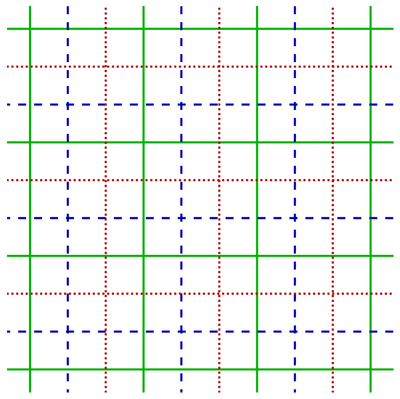
of this range, we still have a high chance of at least one grid cell detecting the event. Figure 6.1 visualizes the geographic grid tokens produced for a location in Washington, DC.

This approach is an improvement over both the Z-curve approach used by Wang et al. [191] and the grid-based approach of Abdelhaq et al. [3]. These approaches would cut e.g. Greenwich into two halves. Our approach does not suffer from such boundary effects. By the pigeonhole principle, we can guarantee that for every point there exists a grid cell such that the location is at least 1/6 of the grid width away from the cell boundary. If a point is closer to two grid boundaries (as visualized in Figure 6.2b), it must be farther away from the boundary in the third grid. In most densely populated areas of the earth (this does not hold at the poles) 1/6 of 2° is 18 to 37 km. The earth has a circumference of 40007–40075 km, thus at the poles $1° \approx 111$ km. Two thirds of an equatorial degree therefore are 37 km. Most populated areas of the earth are within ±60° of the poles (the largest city outside this interval is Helsinki, Finland). At ±60° from the poles, this distance has shrunken to 18 km. The maximum distance within a cell of 2° is 314.5 km. The average grid cell size is about 30 to 50 miles. In other words, we can guarantee that if an event happens within an area of 20 miles diameter, there is at least one grid cell where *all* of the positions end up in. Events farther away than 200 miles of each other are guaranteed to be in different grid cells. In-between of this range, we still have a high chance of at least one grid cell detecting the event.

6.6.2 Tokens based on Administrative Boundaries

For aggregation at coarser resolutions, we employ a different strategy. We use a fast reverse geocoding library[3]. We extracted polygons from OpenStreetMap

[3]`https://github.com/kno10/reversegeocode`


(a) Three overlapping grids

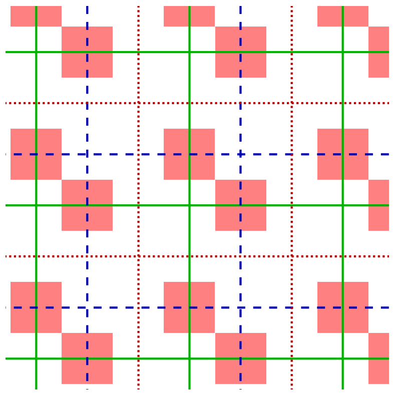

(b) Worst-case areas

Figure 6.2: Pigeonhole principle with 2-dimensional grids: any point that is close to any intersection of the green (solid) and blue (dashed) lines is in the shaded areas, and more than 1/6th of the grid width away from the red (dotted) borders. You can only be close to two borders at any time if they are evenly spaced in 2-dimensional data.

and built a fast lookup table with 0.01 degree resolution, containing about 60.000 regions while using just 30 MB of RAM, and allowing constant-time lookups at a rate of 1.5 million operations per second on a modern CPU. Countries often include their territorial waters, so even tweets from boats close to the shore will be assigned the appropriate region. An overview of the resulting regions is visualized in Figure 6.3. We reverse-geocode each coordinate to a hierarchy of regions, such as *Miami*, *Florida*, *USA* and as seen in Figure 6.4. We use each region returned by the lookup as a geographic token for our process; and this hierarchy allows us to detect events at different levels such as cities, counties, states and countries.[4] Table 7.1 lists the most popular regions in this hierarchy to demonstrate geographic inequality of the data set.

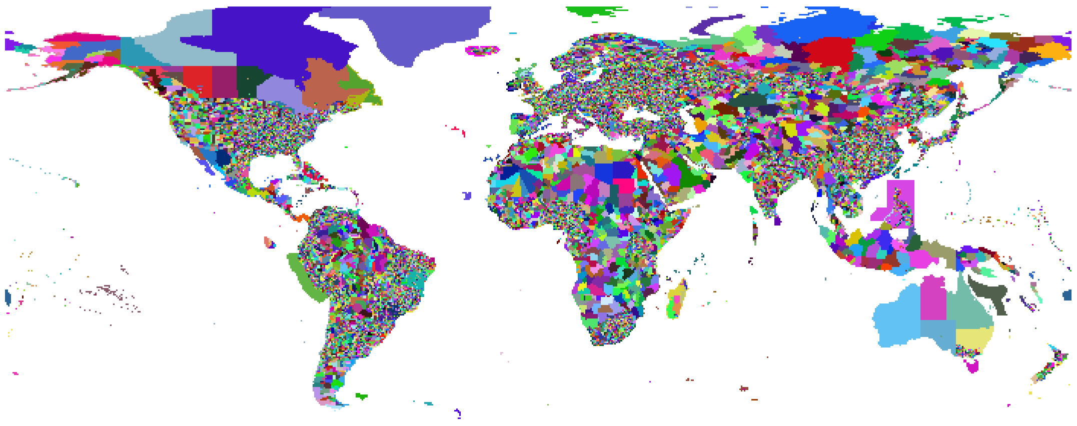


Figure 6.3: Reverse Geocoding with OpenStreetMap Data

[4]OpenStreetMap currently does not provide polygons for continents and oceans.

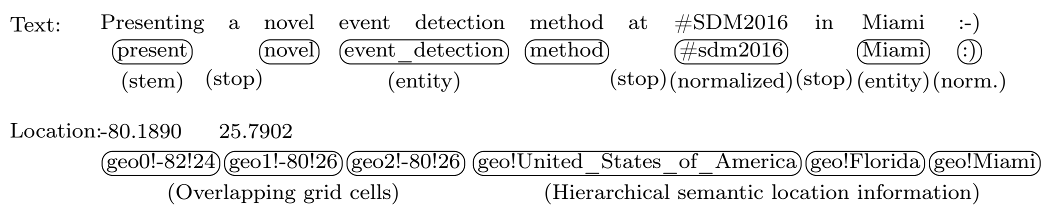


Figure 6.4: Tweet tokenization and generation of geo tokens of an example tweet

6.7 Incorporate Geographical Information

In order to integrate textual data and geographic information, we map both into a sequence of tokens. Textual data is processed by Porter stemming, stop word removal, unification of emoji and hashtags, and a simple entity extraction approach trained on Wikipedia.[5] This provides us with a scalable method that avoids boundary effects and retains enough data to be able to achieve statistical significance. Figure 6.4 shows the tokenization process for an example tweet. After tokenization, documents are logically represented as a set of tokens that either correspond to a word token w or a location token l.

6.8 Significance of Location-Events

Given a word token w and a location token l we use our statistical model introduced in Section 5.4.1 to measure the significance of the event. Computing these exact statistics for every pair (w, l) naively is infeasible due to the large number of different words and locations occurring in a fast-flowing social media stream. As discussed in Section 5.4, we developed an approach derived from Count-min sketches to efficiently maintain these statistics for huge data sets via hashing. An important benefit of our approach over prior work such as GeoScope is the normalization with respect to demographic and geographic differences. The observed counts are standardized by subtracting the expected rate, and normalized by the (exponentially weighted) standard deviation. The resulting z-score is a scale-free factor measuring how unusual the observed frequency is. This allows the statistic to work for highly populated areas as well as much less populated areas: in New York the expected rate will be much higher, than in Twitter-agnostic Germany. As seen in Table 7.1, cities like Istanbul, New York City, Tokyo and London have many more tweets than all of Germany. An approach that does not take local tweet density into account would be unable to detect an event in a city in Germany. This is a major improvement over e.g. GeoScope which effectively only monitors the globally most popular cities, and the approach used by Twitter for their trending topics.

[5] A sequence of words that is frequently used to link to the same article on Wikipedia is considered an entity. Source code and data is available on GitHub: We use the data available at: `https://github.com/kno10/WikipediaEntities`

6.9 Updating the Moving Averages

To compute the exponentially-weighted moving averages for each pair (w, l), we need the previous value (stored in a SigniTrend sketch), the aging factor constant α and the current frequency of each pair. To obtain a reliable frequency estimate, we need to aggregate the data over a large enough time window to not exhibit random fluctuations, which would cause false alerts and a too high variance. Experimentally 15–60 minutes work well, containing about 100,000 usable (non-spam, non-duplicate, geotagged) documents. The statistics are slow-moving averages, therefore this update rate is sufficient. A delay of one hour however is not acceptable to detect "breaking news". Therefore, we split the process into two parts: a batch interval (1 minute) for event detection and the "epoch" interval (15–60 minutes) for statistics updates.

In the main batch interval of 1 minute, Tweets are tokenized, stemmed, spam and duplicates are removed. We can afford to count all frequencies in this batch due to its small size. This data is then continuously fed into a count-min *aggregation sketch*, and at the end of the epoch, the table with the moving averages is updated, and we can start a new sketch for the new epoch. The old sketch is now used as *detection sketch* during the next epoch. We continue to update the counts as in the aggregation sketch, but since it is pre-filled with the counts of the previous epoch, we can obtain a less noisy estimate of the average frequency, because it contains both the new data and that of the previous epoch. In particular during the first minutes of an epoch, we can obtain much more reliable frequency estimates this way and reduce the amount of false positive alerts substantially. At the end of the next epoch, this sketch is discarded and replaced with the next aggregation sketch. The overall process is summarized in Algorithm 3. By using moving averages, we also do not need to remove individual tweets from our statistics but the data "disappears" due to exponential weighting. Other approaches such as GeoScope [41] have to maintain a buffer storing the most recent tweets to be able to remove them from their counts when they have expired. In our approach, data expiry is solved by the exponential weighting of the averages. In addition to the tables for counting, moving average, and moving standard deviation employed by our Event Detection algorithm from Chapter 5, we also maintain a threshold table which contains the alerting thresholds and the last reported values ν. The 1-minute batch interval is used for counting, and the resulting frequencies (aggregated over the active epoch) are compared to these thresholds. If the observed count exceeds the previously reported value by a significance threshold τ, the event will immediately be reported. Every time it exceeds the previously reported value by another τ standard deviations, it will be reported again. At the end of each epoch, the moving average table is updated—the longer update interval yields more reliable frequency estimates—and the last reported value ν is also exponentially decreased to allow recurrent events.

Algorithm 3: Event detection algorithm

```
1  *aggregation sketch* ← new count-min sketch
2  *detection sketch* ← new count-min sketch
3  *threshold table* ← new threshold sketch
4  *statistics table* ← new SigniTrend sketch
5
6  while new batch $D$ do
7      foreach document $d$ in batch $D$ do
8          Tokenize text of document $d$
9          Add geo-tokens for document $d$
10         foreach word $w$ or pair $(w, l)$ do
11             Compute $h$ hash codes
12             Update *aggregation sketch*
13             Update *detection sketch*
14         end
15     end
16
       /* See  Figure 6.5b for details                                */
17     Compare *detection sketch* and *threshold table* Report detected events
18
19     if end of epoch interval then
20         Update *statistics table* using *aggregation sketch*
21         (see Figure 6.5c)
22         *detection sketch* ← *aggregation sketch*
23         *aggregation sketch* ← new count-min sketch
24         Expire old events from *threshold table*
25     end
26 end
```

6.10 Significance Computation

Given a word token w and a location token l we use a classic model from statistics to measure the significance: Let $f_t(w,l)$ be the relative frequency of this pair of tokens within the documents $D_t = \{d_1, \ldots, d_n\}$ at time t, i.e.

$$f_t(w,l) := \frac{|\{w \in d \wedge l \in d \mid d \in D_t\}|}{|D_t|}$$

then we can use the series of previous values $f_1, \ldots, f_{t-1}$ to compute an estimated value and a standard deviation. To facilitate aging of the data and to avoid having to store all previous values, we employ the exponentially weighted moving average ($EWMA[f(w,l)]$) and moving standard deviation ($EWMVar[f(w,l)]$). With these estimates, we can compute the z-score of the frequency:

$$z_t(w,l) := \frac{f_t(w,l) - EWMA[f(w,l)]}{\sqrt{EWMVar[f(w,l)]}}$$

To avoid instability if a pair (w, l) was not seen often before, we employ a bias term β as introduced in [163]:

$$z_t(w,l) := \frac{f_t(w,l) - \max\{EWMA[f(w,l)], \beta\}}{\sqrt{EWMVar[f(w,l)]} + \beta}$$

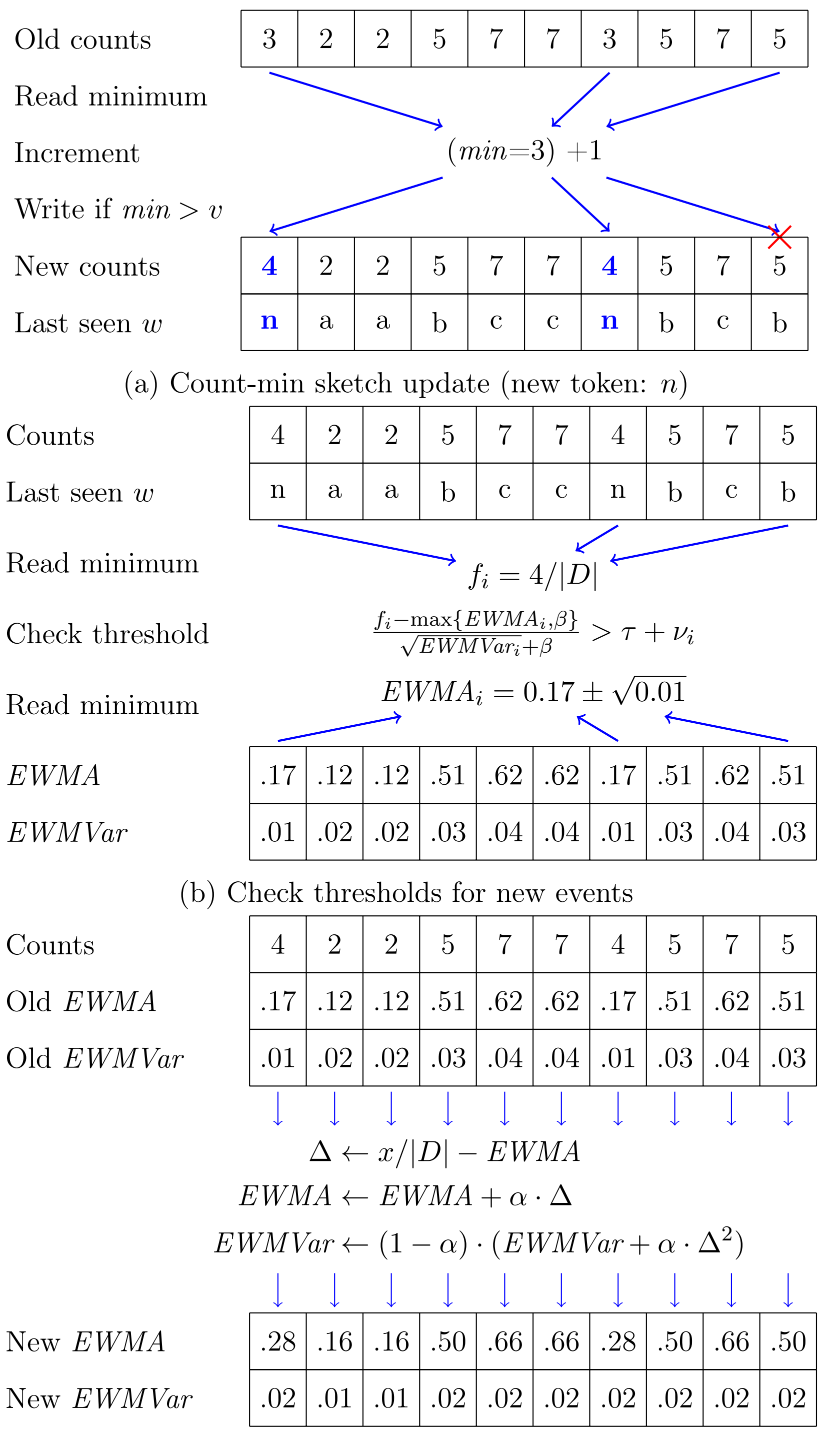
The term β is a Laplace-style smoothing term motivated by the assumption that there might have been $\beta \cdot |D|$ documents that contained the term, but which have not been observed due to incomplete data. For Twitter, the suggested value for this term is $\beta = 10/|D|$: intuitively we consider 10 occurrences to be an observation by chance.

6.11 Hash Table Update Process

In this section we show how we extend our hash table update process (as introduced in Section 5.4) to be able to handle location data as well as text data properly. For every document, each word and word-location pair (w, l) is hashed using h different hash functions into a (simplified: 1-dimensional) Count-min sketch. The minimum value of the hash buckets yields the current approximate count, which is then incremented and updated in every bucket that had the minimum value (a bucket that had a higher value has a collision with a more frequent term; this is the same as writing only if the new value is larger than the previous value). This procedure is visualized in and can be distributed on multiple hosts if necessary. As this step needs to be performed for every document, it needs to be very fast. If we update a value, we also cache the last word or word-location-pair (w, l) for this bucket for reporting. Since this is at most one per bucket, we have an upper limit on the number of candidates.

We use our batch interval of 1 minute to check alerts, a timing interval present throughout our processing toolchain for efficiency reasons. We store cached values of $\sqrt{EWMVar}$ and the last reported threshold ν_i for each hash bucket i. If the current z-score exceeds the last reported value ν_i by the reporting threshold τ, we report an event using the cached word (if it was reported before, it will thus be re-reported at approximately 2τ, 3τ, ... to track further growth in popularity).

At the end of each epoch, the main statistics table is updated. The observed counts are normalized with the number of documents $|D|$ in the current epoch, and the moving average and variances are updated. The last reported values ν_i are decreased to slowly forget earlier trends. These operations are designed so that they can be vectorized to efficiently update the statistics table, as seen in Figure 6.5c. The main reason to perform these updates in epochs is to have stable aggregate statistics: $f_t(w, l)$ will only be reliable when we have seen enough documents $|D|$ since the last epoch.



(a) Count-min sketch update (new token: n)

(b) Check thresholds for new events

(c) Vectorized statistics table update

Figure 6.5: Hash table maintenance

Chapter 7

Data Sets

For our experiments we focused on three data sets that exhibit very different characteristics, as it can be seen from Table 7.2 and Table 7.3. The first data set, consisting of news articles, contains much fewer documents, but the documents obviously are much longer than those of the Twitter data set. The longer paragraphs of the news articles then yield even more word pairs than the Twitter data. But there is also a more subtle difference than the size. While Twitter contained 25 million unique tokens, 99% of these occurred less than 14 times in the data set. In the news data, the vocabulary was much more evenly used, with the top 1% of words occurring 3476 times or more.

7.1 News Articles

We used a web crawler to index news articles from popular international news agencies such as Reuters and Bloomberg as our first data source. We limited the index process to the year 2013 plus a window of 10 days. As these news contents have restrictive copyright requirements, we must not store the full documents, but only use the data to perform trend detection and construct a search index for refinement; but we will be able to direct the user to the full articles on the publisher's website as desired.

Nevertheless, the news article corpus is very interesting, because it is actually editorially managed, and of a different nature than Twitter: while Twitter users tend to re-share the exact same text to their followers, news agencies try to avoid duplication. For some trending topics, we do however see "update" documents in this data source. For the use case of a corporate user interested e.g. in financial trends, news agencies may be the more interesting data source than Twitter. Methods such as enBlogue [21] cannot be applied on this data set without modifications, as they rely on hashtags to seed its search process. Our method does not have such restrictions, but monitors all frequent word pairs.

7.2 Stack Overflow

Stack Overflow is a Q&A website, which publishes its data under a creative commons license.[1] We analyzed the contents of the main website, years 2010 to 2013, totaling to 5.9 million questions. There were few significant trends in this data set, corresponding to important software releases (OS X Mavericks, OS X Mountain Lion, iOS6, iOS7, TypeScript), the `facebook.stackoverflow.com` launch and holidays (New Years, Christmas), so we omit details for brevity (included in the online demonstration).

7.3 Twitter

Twitter data is even noisier and larger in volume, but it is also prone to bias due to its non-representative user base: in particular teen idols make up for a large share of Twitter traffic. We used Twitter's public streaming API (approximately 1% of tweets) to process about 279 million tweets over a period of 114 days. For analysis purposes we only used English language tweets, removed all retweets (about 10%) and used a simple near-duplicate detector to remove obvious spam and bulk (about 1.5%), which yields 94 million tweets to analyze. When including *#hashtags* and *@usermentions* as tokens, the results were dominated by teen idols, which are massively "spammed" by their fans. But as these tweets include few other words (except stop words such as *ilysim*) they become essentially empty when skipping these tags. Without *#hashtags* and *@usermentions*, the analysis becomes more focused on the textual content, and thus the results became much more interesting (as seen in Table 8.2).

7.3.1 Geographic Distribution

For our Geo-spatial Event Detection experiments we additionally collected tweets from Twitters `streaming/track` endpoint with `locations` parameter. Our data set contains 5–6 million tweets per day (slightly more than the popular `statuses/sample` endpoint; no retweets, and all tweets are geo-tagged). However, it contains a much higher rate of geo-tagged tweets, and we estimate that we might be receiving up to 1/3 of the total geo-tagged tweets by interpolation from the sample. The data period is September 10, 2014 to February 19, 2015. Due to the Twitter terms of service, we are not allowed to make this data set available for download. Table 7.1 gives the most frequent locations in the data set, both in millions of tweets and in the relative share of the total data set. Regions such as Germany and Berlin have an unusually low Twitter adoption rate, while other countries such as Turkey are overrepresented. This demonstrates the need to look for relative increase of volume, and to use geographic location for normalization.

[1] `https://archive.org/details/stackexchange`

Table 7.1: Geographic Distribution of Twitter Data. (September 10 2014 to February 19 2015)

Region	Mil.	Share
U. S. A.	287.7	25.4%
Brazil	165.6	14.6%
Argentina	73.6	6.5%
Indonesia	72.0	6.4%
Turkey	59.3	5.2%
Japan	52.4	4.6%
United Kingdom	49.3	4.4%
São Paulo	40.6	3.6%
England	40.3	3.6%
California	38.6	3.4%
Rio de Janeiro	35.9	3.2%
Spain	34.8	3.1%
Buenos Aires	33.9	3.0%
Philippines	32.1	2.8%
France	31.8	2.8%
Malaysia	31.3	2.8%
Texas	31.2	2.8%
Marmara Region	30.1	2.7%
Istanbul	20.3	1.8%
Rio Grande do Sul	20.0	1.8%
Thailand	19.4	1.7%

Region	Mil.	Share
Russian Fed.	18.2	1.6%
Mexico	17.6	1.6%
Florida	17.4	1.5%
New York	17.3	1.5%
Kanto (Japan)	17.2	1.5%
West Java (Ind.)	16.9	1.5%
Saudi Arabia	15.4	1.4%
Colombia	14.1	1.2%
Selangor (Mal.)	14.1	1.2%
Ohio	13.6	1.2%
Kinki (Japan)	13.5	1.2%
Santa Catarina	13.1	1.2%
Los Angeles	12.8	1.1%
Ile-de-France	12.5	1.1%
Minas Gerais (B)	12.3	1.1%
Porto Alegre (B)	12.2	1.1%
Manila (Phil.)	11.9	1.1%
Jakarta (Indon.)	11.6	1.0%
Pennsylvania	11.3	1.0%
Aegean Region	10.6	0.9%
Italy	10.1	0.9%

Selected further regions < 1%:

Region	Mil.	Share
Canada	9.4	0.83%
Portugal	9.3	0.83%
Uruguay	9.0	0.80%
London	7.6	0.67%
New York City	7.5	0.66%
Tokyo	7.4	0.66%
Chile	7.0	0.62%
Bangkok	6.8	0.60%
Los Angeles (City)	6.3	0.6%
Scotland	5.5	0.48%
The Netherlands	4.8	0.43%

Region	Mil.	Share
India	4.8	0.43%
Egypt	4.2	0.37%
Australia	4.0	0.35%
Ukraine	3.8	0.33%
Germany	3.5	0.31%
Nigeria	2.6	0.23%
Pakistan	1.6	0.14%
China	1.2	0.11%
Berlin	0.5	0.05%
Vietnam	0.2	0.02%
Iran	0.2	0.01%
Bangladesh	0.1	0.01%

Table 7.2: Data set statistics after stop word removal

Data	Documents	Paragraphs	Unique Words	Total Words	Unique Pairs	Total Pairs
News	424,704	5,867,457	300,141	56,661,782	71,289,359	660,430,059
Twitter	94,127,149	94,127,149	25,581,022	245,140,695	179,105,233	473,871,456
StackOverflow	5,932,320	30,423,831	2,040,932	138,205,636	91,460,397	545,570,530

Table 7.3: Vocabulary statistics after stop word removal

	Word Frequency			Pair Frequency		
Data	Median	90% Quantile	99% Quantile	Median	90% Quantile	99% Quantile
News	3	81	3476	2	18	124
Twitter	1	at 92.9: 2	14	1	at 91.3: 4	29
StackOverflow	1	at 90.1: 7	185	1	at 90.9: 5	73

Chapter 8

Experiments

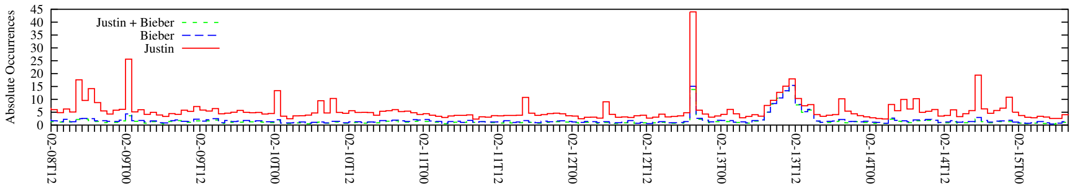
In this chapter we demonstrate the scalability of our system as well as its ability to detect statistically significant trends in real data sets, without the need to reduce the candidate set. Further, we will show how the incorporation of location data can improve results by detecting additional local-specific events that would not have been found otherwise. For all data sets, we employed best practises for text mining such as tokenization and language-specific stemming as introduced in Section 3.6. We also removed a standard set of English stop words as well as domain specific stop words (e.g. *retweet* for Twitter) for efficiency.

The implementation of the main analysis (not including the web crawler and preprocessing) was done in Java using a single thread and Apache Lucene[1] as backing index (for details see Section 8.2). To save disk space and comply with copyright requirements, we did not store complete documents for the news data set, but only inverted lists, the URL and the headline. For Twitter data, we used the original tweet as headline.

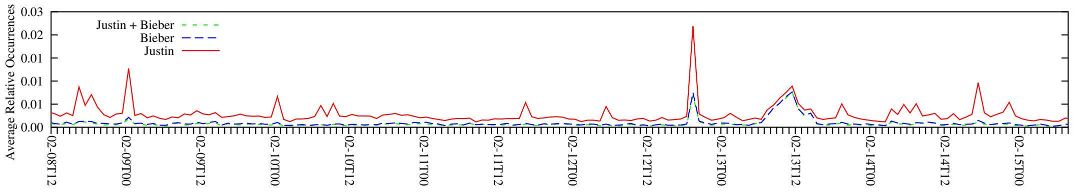
8.1 Manual Analysis of Real World Trends

As previously discussed in Section 3.9 of Part I, there exists no ground truth with labeled events. Thus we start our experiments chapter with a manual analysis: Table 8.1 examines the top 50 most significant trends in the news data set for the year 2013. For brevity, we omitted economic and sport news outside the top 10. As we can clearly see, finance and sports dominate the results due to their extensive coverage in these data sources. In particular, "game days" of sports leagues trend every week due to some new word pair combinations. This may require future work to automatically classify the detected trends into categories such as sports. Nevertheless, a number of events also made it into the top 40, such as the Boston Marathon bombing discussed before, the Algeria gas plant attack and Syria's use of chemical weapons. In Table 8.2 we discuss the top 40 trends detected in the Twitter data set.

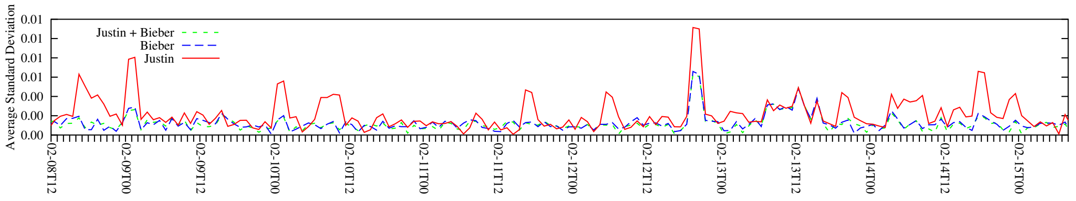
[1]Open-Source, `https://lucene.apache.org/`



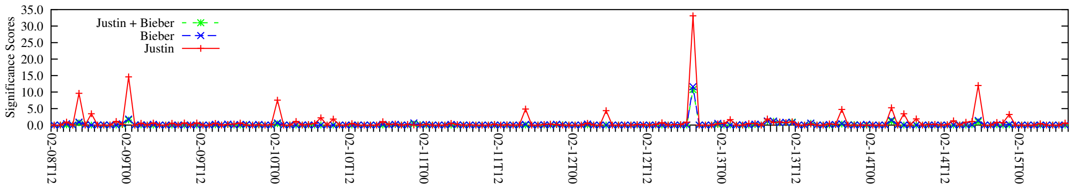
(a) Absolute mentionings of Justin Bieber



(b) Exponentially weighted moving average of Justin Bieber mentionings



(c) Exponentially weighted moving variance for Justin Bieber mentionings



(d) Significance scores of Justin Bieber

Figure 8.1: Even Justin Bieber can trend on Twitter – referring to a Bieber look-alike kissing a man.

Table 8.1: Excerpt of top 50 trends on news data set 2013 (dominated by economy, sports and politics)

σ	Date	Keywords (excerpt, edited for readability)	Explanation
58	11-18	thomson text summary 3000xtra alert outperform eikon	Reuters artifact – 233 research alerts
13	10-09	janet yellen ben bernank vice barack obama nomin	Obama nominates Yellen as U.S. Fed Chief
10	02-14	heinz buffett gdp hj merger shrank berkshir hathaway warren	Berkshire Hathaway, 3G Capital buy Heinz
9.7	07-03	turmoil armi unrest egyptian lisbon egypt mursi portug bailout	Yen rises because of turmoil in Egypt, Portugal
9.4	04-19	boston search bomb suspect marathon manhunt	Boston Marathon suspect manhunt
9.3	04-15	boston explos marathon	Boston Marathon bombing
9.3	01-28	durabl caterpillar pend	Four-month high oil, strong durable goods
8.8	09-19	inning era	Baseball end of season reports
8.8	02-26	ben bernank testimoni defend deadlock stalem	Bernanke defends bond-buying, Italian stalemate
8.6	07-11	ben bernank minut accommod forese dovish	Bernanke reassures euro bonds markets
8.6	04-16	finish sharp metal boston marathon rebound terror bomb explos injur	Boston Marathon attack details
8.6	03-25	rescu cyprus bailout eurogroup guarante relief dutch jeroen dijsselbloem	EU cyprus bailout
8.5	01-17	hostag desert milit algeria algerian	Algerian gas plant attack
8.5	09-03	finnish wireless handset nokia smartphon microsoft	Microsoft buys Nokia's handset business
8.3	01-14	ben bernanke van charl	
8.3	01-23	davo forum switzerland cameron referendum	Cameron promises referendum to leave EU
8.0	09-21	player singl score pitcher roster mike lhp retroact	

Continued on next page

Table 8.1 – continued from previous page

σ	Date	Keywords (excerpt, edited for readability	Explanation
8.0	07-30	uralkali potash	Potash sector takeover
7.9	09-28	score visit rank touchdown offens extra junior vitori	Football
7.8	01-25	repay ifo ecb	ECB announces early repay
7.8	11-24	atom reactor iranian geneva enrich breakthrough lift uranium	Breakthrough on Iran nuclear activity
7.8	08-26	chemic weapon secretari holiday durabl	Kerry comments on Syria over chemical weapons
7.7	05-21	oklahoma moor tornado	Moore, Oklahoma hit by deadly tornados
7.6	09-23	merkel angela coalit victori william dudley shutdown	German elections success for Angela Merkel
7.3	10-01	deadlock shutdown midnight began trillion unpaid barack obama	U.S. government partial shutdown
7.2	08-27	syrian chemic weapon strike assad tension kerri secretari	Escalations in Syria
6.9	04-20	dzhokhar tamerlan tsarnaev brother suspect captur dead watertown injur	Boston Marathon suspects captured
6.8	07-24	appl iphon smartphon markit flash beat faster	Surge in iPhone sales
6.8	12-06	nelson mandela die apartheid african africa	Nelson Mandela died
6.7	02-15	moscow communiqu	G20 finance ministers in Moscow
6.7	04-08	thatcher iron margaret	Margaret Thatcher died
6.6	07-12	airport dreamlin ethiopian heathrow boe	Boeing Dreamliner catches fire

Table 8.2: Top 40 trends on Twitter data (dominated by celebrities and teen idols)

σ	Date	Keywords	Explanation
174	03-06	boosi releas jail	Rapper Lil Boosie released from jail early
154	05-28	rip author poet inspir angelou maya	Civil rights activist Dr. Maya Angelou died
127	05-12	elev jayz attack jay solang beyonc	Solange, Jay Z and Beyonce elevator incident
98	03-03	ellen degener host selfi pizza	Ellen's Oscar Selfie and Pizza
91	02-10	carl rick	The Walking Dead TV series midseason premiere
76	05-22	ewok	Band 5SOS changed its name on Twitter to "Ewok Village"
76	03-21	bracket mercer duke	Mercer surprise win over Duke in March Madness
73	05-24	ronaldo bale gareth	Champions league final
64	02-19	vote #BRITs2014	The 2014 British Phonographic Industry's annual pop music awards
63	04-07	geldof dead rip peach	Peaches Geldof died of heroin
61	04-15	moon eclips lunar blood	Blood moon (lunar eclipse)
60	05-05	shovel	Snow in May + "shovel girl fight" viral video
51	05-24	ramo sergio	Champions league final
51	04-14	zac efron	Zac Efron shirtless at the MTV movie awards
50	03-17	punch pinch green	St. Patricks Day
50	05-06	mimi	Mimi Faust and Nikko announce private video
49	03-17	lizzi	Lizzie in the series The Walking Dead
48	02-19	angela	Talking Angela hoax
47	04-29	ban silver adam donald sterl nba	Donald Sterling banned for life from NBA
46	05-15	sm exo kris	Rumor of Kris (of Exo-M) parting
44	02-14	valentin	Valentines Day greetings
43	05-10	austria	Austria wins the Eurovision song contest

Continued on next page

Table 8.2 – continued from previous page

σ	Date	Keywords	Explanation
43	03-03	arm bradley longer cooper	Oscar selfie: “If only Bradley’s arm was longer. Best photo ever”
43	02-24	rip harold ghostbust rami	Actor Harold Ramis died
42	04-10	sibl nation	National siblings day
42	03-09	arsenal wigan	Arsenal vs. Wigan Athletic in FA cup
41	04-13	million matt	Matthew Espinosa follow spree (#mattTo1Mil)
41	03-17	earthquak	4.4 earthquake strongly felt in Los Angeles
41	05-09	billionair dre dr beat billion appl	Apple buys Dr. Dre
40	05-10	eurovis	Eurovision Song Contest
38	05-06	bambi	Love & Hip Hop Atlanta Premiere with Bambi
38	03-19	ezra dead	Ezra gets shot in series “Pretty little liars”
38	04-16	bale gareth goal	Winning goal in Copa del Rey final
38	03-03	actor oscar mcconaughey matthew leo leonardo dicaprio	No Oscar for Leonardo di Caprio, again
37	05-06	scrappi	Love & Hip Hop Atlanta Premiere with Scrappy
36	05-09	cleveland johnni manziel dalla footbal draft brown	Johnny Manziel draft in NFL
35	04-30	rip bob actor hoskin	Actor Bob Hoskins died
35	04-26	racist donald clipper sterl owner	Donald Sterlin accused of racism (see above!)
35	02-08	arsenal	Liverpool vs. Arsenal 5-1 football match at Anfield
35	03-18	allison	Allison in TV series “teen wolf”
34	04-02	tsunami chile magnitud earth-quak	Chile 8.2 magnitude earthquake
34	05-21	cav	Cleveland Cavaliers win NBA draft lottery again
34	05-06	joselin	Love & Hip Hop Atlanta Premiere with Joseline Hernandez

Continued on next page

Table 8.2 – continued from previous page

σ	Date	Keywords	Explanation
34	03-14	pie pi	π day (3.14)
32	02-12	justin bieber	Justin Bieber commenting of a look-alike kissing a man (see Figure 8.1)
30	02-15	disini download flappi	iPhone and Android game “Flappy Bird” clones hit the app stores
28	02-20	canada	Sochi: Canada wins women’s hockey gold
21	02-18	luke 5sos hem	Luke Hemmings (nickname: Luke5SOS) teen idol
17	02-17	mvp	Kyrie Irving gets MVP (Most Valuable Player) Award in NBA
16	02-15	earthquak	Earthquake of magnitude 4.1 in the Pacific Ocean near Canada
15	02-16	drake	Kanye West defends Drake against Rolling Stone Magazine in new Rant
14	02-10	michonn	start of Season 4 of the TV series “The Walking Dead”
13	02-19	whatsapp	Facebook buys WhatsApp for $19 billion

All of these events are easily verifiable with internet search. Detected events consist of single words, up to almost complete sentences by Twitter standards. Despite not using retweets, many topics such as the Oscar selfie (the most retweeted tweet so far) still surfaced, because a substantial amount of users add a reply text to their tweet.

We were surprised to find Justin Bieber trending – but detailed analysis (see Figure 8.1) showed that our method was right: apparently an old image resurfaced of someone looking somewhat like Justin Bieber kissing a man. This led Justin Bieber to comment on "some of the rumors out there", and thousands of his fans replied. The wider and smaller peak in Figure 8.1a on February 13 is the media coverage of this Twitter fad.

To ensure that real world trends are found, we manually evaluate our results against the top 10 lists from Google Trends[2]. Also, we include the "Most Searched News Stories" from Microsoft Bing[3] as seen in Table 8.3.

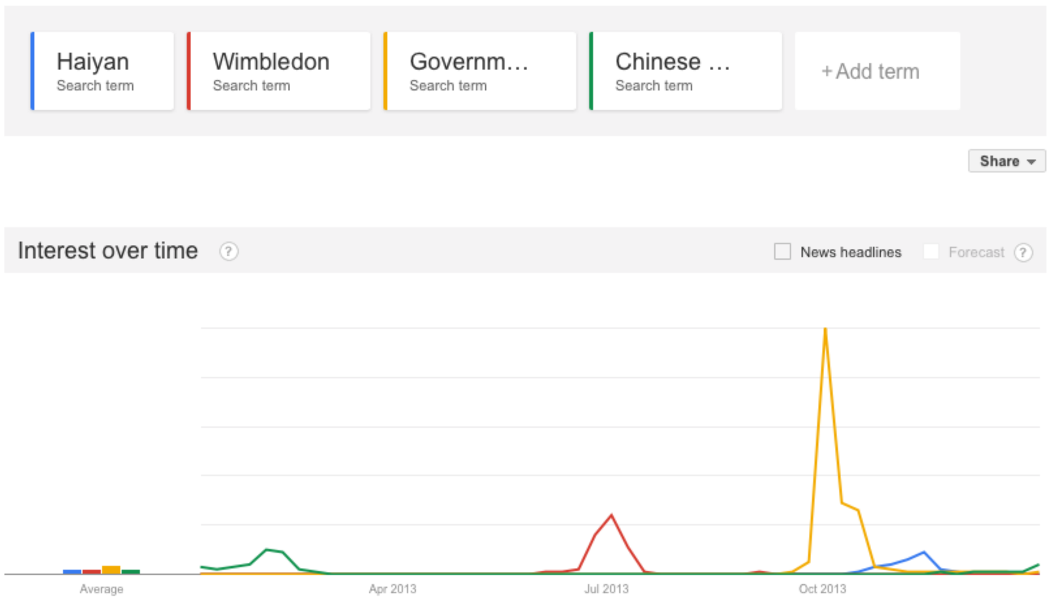


Figure 8.2: Google Trends chart for some of the top 10 events in 2013 for Typhoon Haiyan, Wimbledon, Government Shutdown and Chinese New Year (source: `http://www.google.com/trends`)

To demonstrate that our Event Detection algorithm is also capable of revealing details and sub-events from major events we analysed the 8th July 2014 where the famous FIFA World Cup football match of Germany versus Brazil took place. Table 8.4 shows all terms related to football player names of both teams and their first significant occurrence with $\sigma >= 3$ and Table 8.5 the top scored events at that day.

[2]`http://www.google.com/trends/topcharts`
[3]`http://www.bing.com/blogs/site_blogs/b/search/archive/2013/12/01/eoy.aspx`

Table 8.3: Top 10 "Most Searched News Stories" from Microsoft Bing in 2013

rank	description
1	Royal Baby Born
2	Boston Marathon Bombing
3	Cleveland Kidnapping
4	George Zimmerman Trial
5	Gun Rights
6	Tesla
7	Syria
8	Anthony Weiner
9	Oil Prices
10	Fiscal Cliff

Table 8.4: Terms related to names of football players during the FIFA World Cup football match Germany vs. Brazil in 2014. Each row represents a snapshot of the first observation that caused a score $\sigma >= 3$.

Event Terms	UTC Time	Expected	StdDev	Have	Score
victor	10:40:05	18.6	0.7	31.0	3.2
mertesacker	11:34:14	8.3	0.6	20.0	3.2
fred, fred's	13:58:45	25.9	1.1	39.0	3.0
lahm	18:52:31	9.8	0.6	21.0	3.0
schweinsteiger	18:52:37	7.8	0.4	19.0	3.2
boateng	18:52:37	6.7	0.5	18.0	3.2
khedira	18:53:03	9.6	0.5	21.0	3.2
hummels	18:53:53	26.9	1.4	41.0	3.0
kroos, tonikroos	18:56:05	24.8	1.8	40.0	3.0
klose	18:56:28	28.4	1.3	43.0	3.2
neuer, manuel_neuer	18:57:59	34.3	1.7	50.0	3.1
müller	18:58:53	16.0	0.9	29.0	3.2
bernard	18:59:34	9.0	0.6	21.0	3.3
özil	19:12:18	31.0	1.5	46.0	3.1
willian	19:14:00	10.8	0.5	22.0	3.1
hulk	19:43:47	28.0	0.9	41.0	3.1
luiz, luiz's	19:58:06	39.0	1.6	54.0	3.0
marcelo	20:18:10	25.8	1.1	39.0	3.0
alves	20:23:05	6.0	0.3	17.0	3.3
david	20:28:06	183.4	7.3	220.0	3.0
dante	20:30:10	19.0	0.7	31.0	3.1
oscar, oscar11	20:43:38	67.8	2.1	86.0	3.1
fernandinho	20:50:11	4.0	0.2	14.0	3.1
gustavo	20:50:17	5.0	0.2	15.0	3.0
paulinho	21:15:01	13.0	0.5	24.0	3.0
schürrle	21:25:12	4.0	0.2	14.0	3.1
draxler	21:31:53	7.0	0.3	18.0	3.2
podolski10	22:23:44	31.7	2.0	48.0	3.0

Table 8.5: Top Trends during the FIFA World Cup Football match Germany vs. Brazil in 2014

Term	Expected (EWMA)	StdDev	Have	Score
alfredo	18.62	0.7967671	1201.0	296.8591
brazil	1322.0	50.22544	18784.0	262.8021
wale	19.567549	0.9052597	829.0	197.37753
connor	32.0	0.9856072	874.0	195.55894
brazilvsgermany	131.0	4.9894543	1943.0	194.85014
germans	62.0	2.1735983	1150.0	187.79349
penalties	83.0	2.8322902	1315.0	184.9214
bravsger	84.0	3.2925277	1400.0	184.50682
goal	232.0	4.2230525	1980.0	183.1699
thumbs	390.04	20.97164	5480.0	182.61885
cam	206.0	7.3655663	2437.0	179.54916
brazilian	76.0	2.719195	1229.0	177.95421
germany	1010.0	47.88509	11626.0	174.07533
halftime	4.804388	0.2438697	459.0	137.97313
granger	9.0	0.46838966	481.0	132.64427
tammy	38.22	1.203601	646.0	132.53519
waka	24.40586	1.105165	590.0	130.04485
redeemer	21.0	0.64333415	520.0	129.49825
brazil's	28.0	1.0445048	588.0	129.4946
tylerjblackburn	14.7	0.5626085	487.0	127.318016
silva	77.0	3.3581972	978.0	126.39942
penalty	92.32923	3.9420476	1085.0	126.20825
meek	51.04578	2.3394494	779.0	124.43859
luiz	39.0	1.606864	658.0	123.8777
score	97.0	3.0099914	959.0	123.49586
schurrle	6.0	0.24659236	414.0	123.38987
half	345.0	4.5833983	1705.0	123.262115
brazilians	40.3368	1.7761872	678.0	123.11157
scoring	17.0	0.6683261	488.0	122.70974
embarrassing	45.177216	0.9336876	578.0	121.49759
german	118.0	6.0781302	1357.0	120.78224
worldcup2014	126.0	5.7381234	1312.0	118.62226
neymar	355.0	14.307218	2814.0	117.89684
brazils	10.0	0.40031707	338.0	93.705795
schürrle	4.0	0.19001406	305.0	93.188446
germany's	22.0	0.9485045	410.0	93.07895
shootout	7.0	0.35850522	320.0	91.29343
klose	187.0	6.790881	1218.0	88.415276
anthem	22.878042	0.7083457	366.0	87.15036
cia	21.0	0.9055899	375.0	86.014404
prayforgaza	22.0	1.0513406	386.0	85.21915
footballers	18.982338	0.90885305	368.0	85.15375
brasil	111.0	4.0090485	801.0	84.98533
ronaldinho	54.0	1.9335893	518.0	84.7707
statue	32.282887	0.984969	397.0	84.6644
goalkeeper	44.236027	2.5766847	548.0	83.695
drawing	109.0	3.7439468	761.0	83.22752
romero	29.0	0.7479012	365.0	83.21155

8.2 Scalability

In this section we demonstrate the effectiveness of our algorithm. Due to the use of hash tables, we were able to run the experiments on a single core without the need to use distributed computation or multiple threads. Our implementation is using Java, and we used a desktop PC with an Intel Core i7-3770 CPU (2.4 GHz Core i7, 8 GB RAM) running Debian Linux and OpenJDK 1.8 with the fastutil and ELKI [162] libraries. Large parts of the implementation were done using low-level data structures to achieve high throughput. Even with taking location data into account we are able to process the twitter stream 400× faster as we will receive the data in a live setting. Therefore, this approach is scalable to the full Twitter "firehose" stream where we still would obtain a performance of $\approx \frac{400}{100} = 4\times$ real-time because our computation time grows only linear with the number of observations (we only need to calculate hashes and update the corresponding hash buckets; all this is done in constant time).

8.2.1 Using only Text Data

For the news data set, the preprocessed data volume was 1.1 GB (2.6 MB per day on average). Analyzing a year of data took 18 minutes, i.e. 2.5-3 seconds on average per day. The refinement index only stores the headlines, for which we need only 745 KB per day, and 305 MB total for a whole year.

The raw 1% Twitter input data was 228 GB compressed, after filtering retweets and non-English tweets, and preprocessing, 6.7 GB remained. Processing Twitter took 1.5 hours; 46 seconds on average per day. This yields a real-time performance of $\approx 1,878\times$ faster than needed for real-time in a live environment. The actual analysis ran in 48 minutes; 25 seconds per day. The inverted index for Twitter occupied 18 GB of disk space, storing about 160 MB per day of Twitter data. By using our effective hashing scheme on the news data set, we were even able to use a Raspberry Pi (700 MHz ARM CPU, 512 MB RAM) to gain a processing speed of 104.1 seconds on average per day ($\approx 830\times$ real-time).

Even when including location data we are able to run the full analysis of one hour of data in 9 seconds on a single core. We could thus process 400× as much data as we will get from the Twitter sample API. This performance is achieved, again, without upgrading the CPU, adding additional cores, or having the overhead of a distributed implementation.

8.2.2 Incorporating Location Data

As a comparison to our algorithm with the inclusion of location data, we implemented GeoScope [41]. We obtained similar results to Budak et al. using their suggested parameter values *topic dominance* $\theta = 0.002$, *location support* Ψ and *location popularity* ϕ both set to $\phi = \Psi = 0.05$. Using hashtags only (as in [41]), we obtained a performance of around 480× real-time, so that processing one day (1,440 minutes) of archived tweets took 3 minutes.

Because concepts like hashtags are likely to be absent in data sources other than Twitter, we also experimented without this restriction and instead process the complete stemmed text of each tweet. Our algorithm was designed to handle *all* tokens. Their individual importance and interestingness is determined by calculating significance scores as discussed in the previous sections. Thus, we examined the real-time performance as well as the total number of reported location-topic instances of GeoScope while dropping the hashtags-restriction and instead process the complete stemmed text of each tweet (so that the input term vector is exactly the same as for our algorithm).

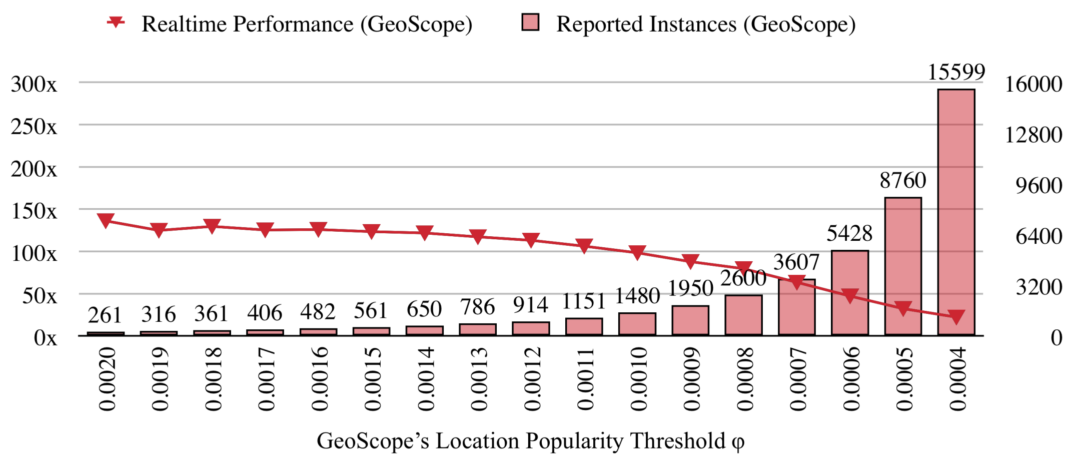
In Figure 8.3a we summarized the real-time performance as well as the total number of reported location-topic instances for a window size of one hour. If we decrease GeoScope's threshold parameters from $\theta = 0.002$ and $\phi = \Psi = 0.02$ in 0.0001 and 0.001 steps. Thereby, the total number of reported instances increases exponentially from 261 to 15,599 whereas the runtime performance decreases substantially from 136× to 32× real-time because the thresholds are less selective.

If a user wants to discover local events tied to geo-locations with low Twitter activity, see Table 7.1 for examples, GeoScope's threshold parameters have to be set to low values to enable tracking for less frequent locations. For example on the day of the Guam earthquake (c.f. Table 8.10) the location *Guam County* ranked only 834th most frequent; thus GeoScope's ϕ parameter needs to be at least $\phi \leq \frac{1}{834} \approx 0.001$ in order to be included. With such low threshold values, the number of reported instances increases to thousands of reported location-topic combinations (c.f. Figure 8.3a). This makes it difficult for the user to discover important topics because GeoScope does not provide a measure of unusual behavior (e.g. trivial topics like *#ElPaso*, *#Milano*, and *#NYC* are reported for El Paso County, Milan, and New York City respectively).

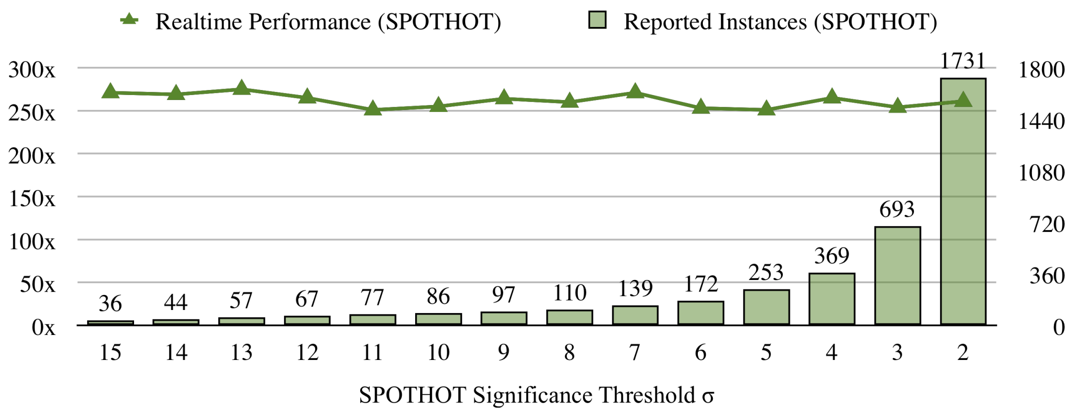
Our algorithm in contrast, provides the intuitive sigma threshold that captures the unexpectedness and allows a more useful ranking of the detected events. At the same time it provides a stable and higher performance with around 260× faster than real-time. There are much fewer statistically significant events, and the performance of this approach does not degrade substantially for reasonable values of the significance threshold σ, as seen in Figure 8.3b.

8.2.3 Distributed Computing

If additional scalability is needed, our algorithm could be implemented in a cluster environment to distribute the workload across nodes of a computing engine like *Hadoop*, *Storm*, *S4*, *Samza*, *Flink*, or *Spark*. This can be achieved because our algorithmic core – hash based counting – is easy to distribute,



(a) GeoScope performance for varying *location popularity thresholds* $\theta = \{0.002, \ldots, 0.0004\}$. *Dominance* and *support* are set to one larger magnitude $\phi = \Psi = 10 \times \theta$ (as seen in GeoScope's own experiments). Left Y-axis: real-time performance (how much faster an archived day of twitter could be analysed) compared against a SpotHot baseline with fixed significance threshold $\sigma = 10$. Right Y-axis: number of reported location-topic instances by GeoScope.



(b) Our performance for increasing significance threshold σ. Left Y-axis: real-time performance. Right Y-axis: Number of reported events per hour.

Figure 8.3: Scalability of GeoScope and our method (SpotHot).

because it is “embarrassingly parallel”[4]. Incoming Tweets can be processed by multiple nodes, and the aggregation tables can be split into multiple slices and this way distributed onto multiple computers. Often, data can additionally be aggregated locally (as done using a “combiner” in MapReduce) and then only these aggregates are transferred to the aggregation nodes.

Additionally, there are a number of easy ways for distributing the load: for example, building the backing index, maintaining the hash tables, and refining

[4]“Embarrassingly parallel” is also called “perfectly parallel”

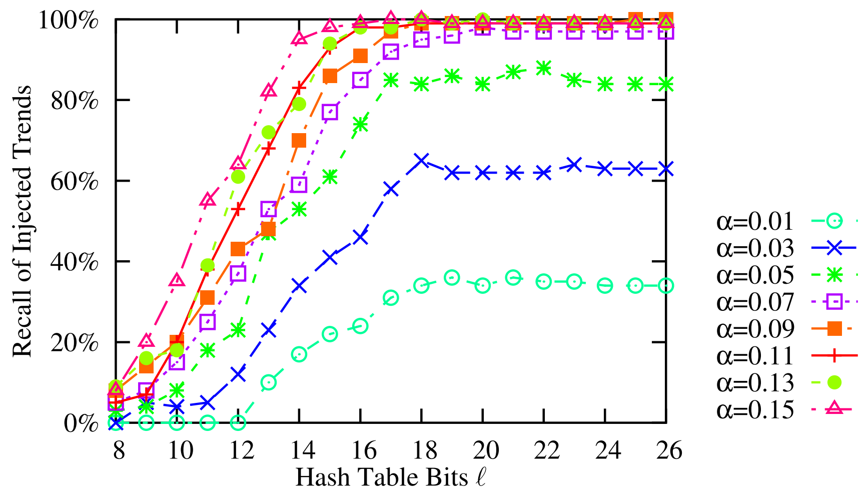


Figure 8.4: Performance with varying hash table size ℓ, $k = 4$

the detected events can easily be distributed on separate systems. As the backing indexes are sharded already, the shards can also be distributed onto different machines trivially.

8.2.4 Hash Table Size

For the hash table sizes, we expect a saturation effect to happen. Too small hash tables will lead to masking and swamping [32, 77]; but once the hash table has become big enough to not have collisions, the *EWMA* estimates are expected to be good. To verify this, we used semi-synthetical data sets derived from the news data sets. Into these, we injected artificial words following a narrow Poisson distribution with $\lambda = 2 \dots 9$ weakened by a constant factor of α. After that we randomly inserted the artificial trends. We modified each document with a probability of:

$$\alpha \cdot pmf_{Poisson}(\lambda, k) = \alpha \frac{\lambda^k}{k!} e^{-\lambda}$$

Due to the randomization of λ, some trends will be easier to spot, others will be harder. Furthermore, both the existing natural trends in the data and injected trends may mask the injected trends. The maximum value of the probability mass function is 0.27, so for $\alpha = 0.15$ about 4% of documents were modified to include the artificial keyword. For $\alpha = 0.01$, the chance of trend injection drops to 0.27%. Figure 8.4 summarizes the results on the injected trends data using $k = 4$ hash functions. As you can see, at $\alpha > 0.05$ the artificial trends are detected reliably, and the hash tables show a typical saturation effect, where the performance no longer increases once we have reached a reasonable size. A hash table of 20 bits requires just 2^{25} bytes of memory (32 Megabytes).

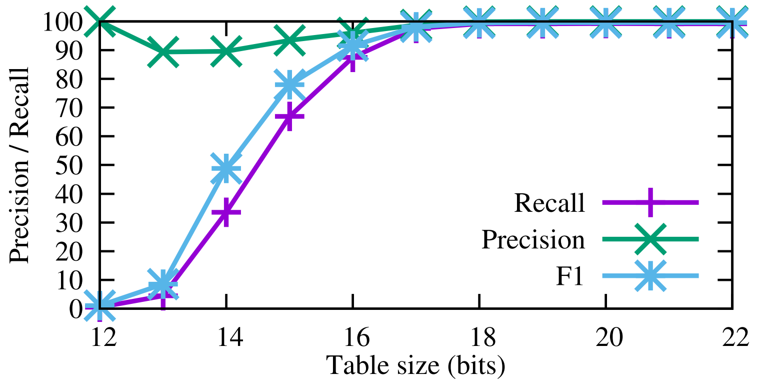


Figure 8.5: Recall and precision compared to exact counting.

8.3 Comparison with Exact Frequencies

To evaluate the approximation quality of our algorithm, we compare to an implementation that uses expensive exact counting on all pairs (old pairs are forgotten if they have not been observed for several hours to reduce memory consumption, but we need 16 GB of RAM nevertheless to be able to keep all candidates in memory).

In Table 8.6 we give recall and precision comparing our approach to this ground truth, using all events with over $\sigma \geq 20$, and consider a match positive if the event is also found by the other method with at least $\sigma \geq 10$ and with at most 1 hour difference. Recall measures how many events found by exact counting are also found by the approximate methods. Precision measures how many of the events found using the approximate method can be confirmed using exact counting.

Table 8.6: Recall and precision compared to exact counting.

Table Bits	Recall	Precision	F_1-Measure	$\sigma \geq 20$
12	0.52%	100%	1.04%	5
13	4.47%	89%	8.51%	47
14	33.56%	90%	48.84%	338
15	66.96%	93%	78.01%	1717
16	87.52%	96%	91.57%	4604
17	97.54%	99%	98.16%	7493
18	99.16%	100%	99.54%	9330
19	99.22%	100%	99.61%	9791
20	99.23%	100%	99.61%	9929

Table 8.5 visualizes these results. Precision remains high (i.e. few incorrect events are reported), but only very few events are detected and the recall is thus very low. The ongoing increase of $\sigma \geq 20$ events are mostly redundant reportings (same event reported multiple times with increasing σ). We can observe a saturation effect on Twitter data at a table size of 18 bits.

Table 8.7: Most Significant Events in their Most Significant Location Each

σ	Time	Word	Location	Explanation
2001.8	2014-10-29 00:59	#voteluantvz	Brazil	Brazilian Music Award 2014
727.8	2014-09-23 02:21	allahımsenbüyüksün	Denizli (Turkey)	Portmanteau used in spam wave
550.1	2015-02-02 01:32	Missy_Elliott	United States of America	Super Bowl Halftime Show
413.5	2014-09-18 21:29	#gala1gh15	Spain	Spanish Big Brother Launch
412.2	2014-11-11 19:29	#murrayftw	Italy	Teen idol triggered follow spree
293.8	2014-10-21 12:05	#tarıkgüneşttyapıyor	Marmara Region	Hashtag used in spam wave
271.2	2015-02-02 02:28	#masterchefgranfinal	Chile	MasterChef Chile final
268.1	2015-01-30 19:28	#سباركيز	Saudi Arabia	Amusement park “Sparky’s”
257.7	2014-11-16 21:44	gemma	United Kingdom	Gemma Collins at jungle camp opening
249.1	2014-10-08 02:56	rosmeri	Argentina	Rosmery González joined Bailando 2014
223.1	2015-01-21 18:51	otortfv	Central Anatolia Region	Keyword used in spam wave
212.7	2014-09-11 18:58	#catalansvote9n	Catalonia	Catalan referendum requests
208.4	2014-12-02 20:00	#cengizhangençtürk	Northern Borders Region	Hashtag used in spam wave
205.3	2015-01-04 15:56	hairul	Malaysia	Hairul Azreen, Fear Factor Malaysia
198.7	2014-12-31 15:49	あけましておめでとうございます	Japan	New Year in Japan
198.5	2015-01-10 20:19	вк	Russian Federation	“Russian Facebook” VK unavailable
179.7	2014-10-04 16:28	#hormonestheseries2	Thailand	Hormones: The Series Season 2
174.7	2014-11-28 21:29	chespirito	Mexico	Comedian “Chespirito” died
160.9	2014-09-21 21:27	#ss5	Portugal	Secret Story 5 Portugal launch
157.3	2014-09-24 01:57	maluma	Colombia	Maluma on The Voice Kids Colombia

8.4 Most Significant Regional Events

In this experiment, we demonstrate that the most significant local events detected, as this shows both the ability to detect trends as well as to locate them geographically. For presentation, we selected the most significant occurrence of each keyword only, and of all locations only the most significant keyword. The top 20 most significant events are shown in Table 8.7. There are four events that we attribute to a highly active Twitter spammer in Turkey,[5] but all others can be attributed to major media events or celebrities. Most of the events in Table 8.7 are detected at country level, which makes sense as they are based on virtual events in TV, and did not happen at the users location.

Overall, we observe that Twitter users mostly comment on what they see on TV and on the internet, and much less on the physical world around them.

8.5 New Year's Eve

We analyzed trending topics on New Year's Eve, and were able to identify New Year greetings in several languages. Interesting patterns emerge if we plot longitude versus time, as seen in Figure 8.6. In the first Figure, the x-axis is geographic, but the y-axis is temporal, so we can see the arrival of the New Year in different time zones. All events with $\sigma \geq 3$ were included, if we were able to identify them as New Year related. To remove visual clutter, we did not include other events, or festive emoji. Several cultural patterns arise in this figure. Chinese New Year is on a different date, and also India does not show a lot of activity. Some vocabulary such as "Silvester" refers to the whole day before New Year. Italians started tweeting "capodanno" in the afternoon, but wish "buon anno" after midnight.

Despite the fact that Twitter is not used much in Germany (see the statistics in Table 7.1), our variance-based approach however can account for this, and was still able to detect some German New Year's wishes, but the English greetings were more popular in Germany on Twitter. Sydney has the first fireworks at 9pm already (for the kids), and we can indeed observe an event "fireworks" in Australia at 10:04 UTC. Throughout the evening, we see New Year mentions, and the event at midnight is rather small compared to other countries. In Québec, we observe more French wishes around GMT than at midnight, but closer inspection revealed that most originate from the same user. In Russia, we can see Yakutsk, Ulan-Ude and Irkutsk, and Novosibirsk celebrate in different time zones prior to the more densely populated areas beginning with Yekaterinburg.

[5] Apparently, this spammer controls thousands of hacked clients and by varying the text contents successfully bypassed both Twitter's and our spam filters. Spam filters require further research.

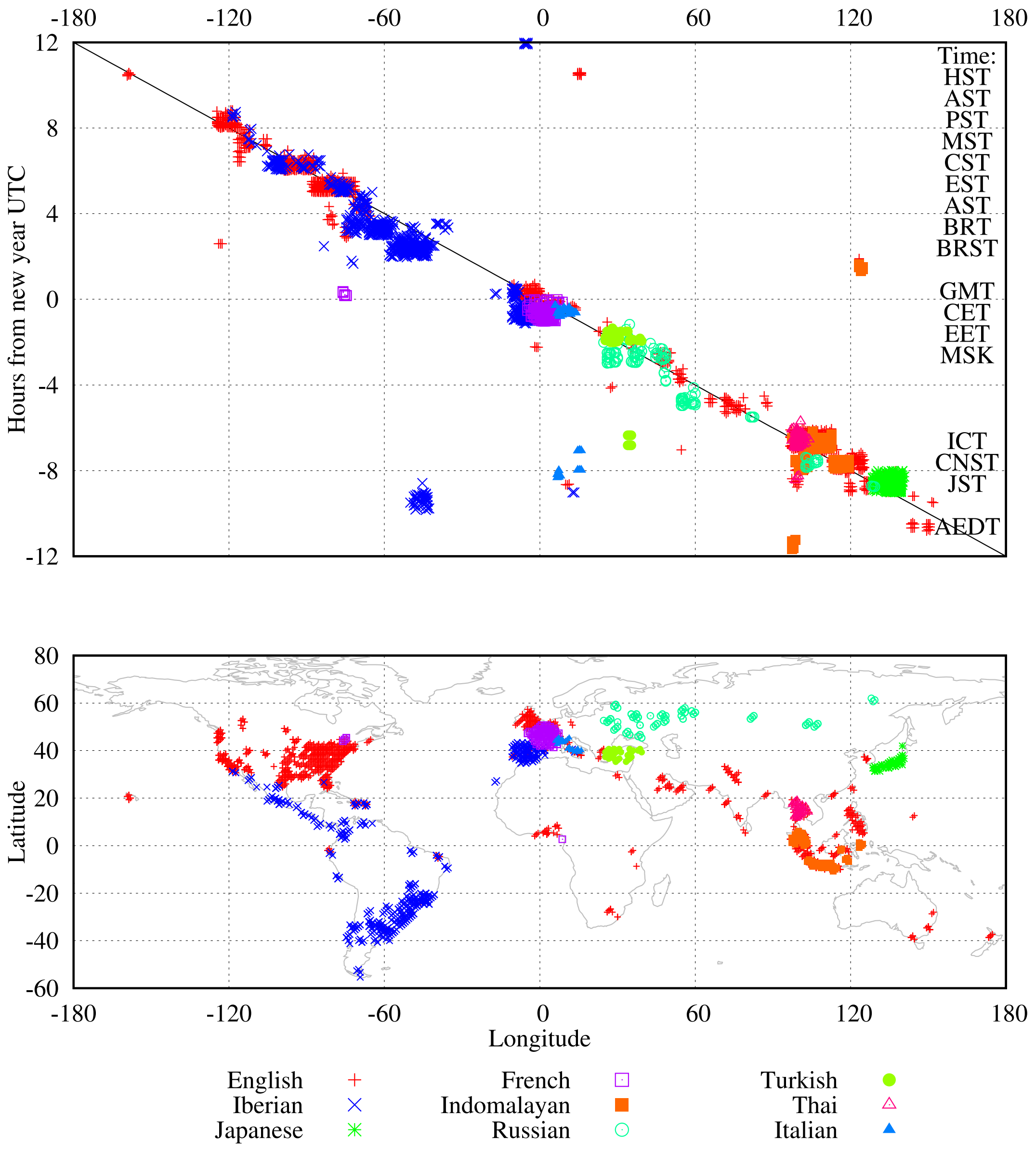


Figure 8.6: New Year around the world at $\sigma \geq 3$

8.6 WikiTimes Events

We use the WikiTimes data set [184] to validate events found by our detection system. Table 8.9 lists events validated by comparing the resulting word cluster (using hierarchical clustering) with Wikipedia headlines for the same day.

Our algorithm was able to identify several important events of contemporary history, and was able to produce both meaningful geographical information as well as related keywords. Only one of these events was detected with a hashtag, indicating that restricting the input data to hashtags—as required for GeoScope—may obscure important events. Named entity recognition however helped in detection, as it can normalize different spellings and abbreviations of names.

8.7 Earthquakes

Natural disasters, such as earthquakes, are of broad interest to the public. It has been proposed to detect them using Twitter [157], thus we evaluate this scenario. The relative frequency of the term "earthquake" regardless of its geographical origin is shown in Figure 8.7a. This yields a noisy signal with a high background activity and many low-significance events. Note that the frequency shown is normalized by the total number of tweets at each corresponding day to avoid seasonal patterns (e.g. more tweets on weekends than on workdays). Once we narrow down the scope to a specific geographic region, we get a much clearer signal. Figure 8.7b shows the frequency only near Guam (in the western Pacific Ocean) within latitude 144 ± 1 and longitude 13 ± 1, where we can observe two events. To validate our observations, we use metadata from the *United States Geological Survey's (USGS) Earthquake Hazards Program*[6] as our validation data source for earthquake events. The first peak in Figure 8.7b at September 17, 2014 refers to a strong earthquake with a magnitude of 6.7 located 47km northwest of Hagatna (Guam). This earthquake was strong enough, that USGS included it in their *Significant Earthquake Archive.*[7] The second, smaller peak on October 29, 2014 refers to a small earthquake 36km southeast of Hagatna with a lower magnitude of 4.7. In Figure 8.7d we plotted the frequency for earthquake mentions around Dallas, Texas within latitude -97 ± 1 and longitude 33 ± 1.

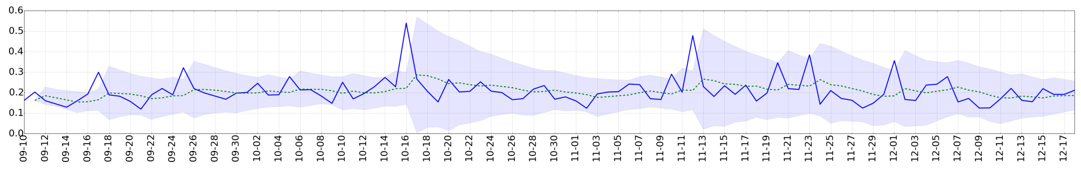
The total number of significant earthquakes reported by the USGS during the time of our Twitter crawl is 30 (ranging from the first earthquake reported on September 17, 2014 to February 13, 2015). By exact counting the tweets corresponding to these events, we found that 19 of them received less than 10 mentions since they happened far away from cities. As this is frequency is too low to be statistically significant we thus excluded them from evaluation.

[6]`http://www.usgs.gov`

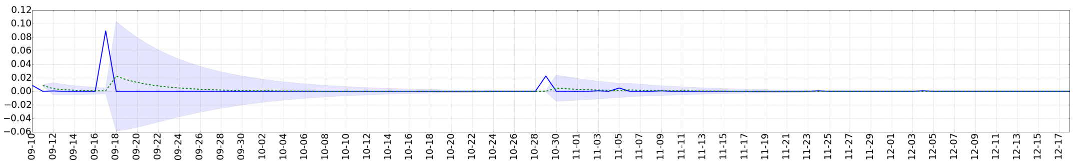
[7]`http://earthquake.usgs.gov/earthquakes/`

Table 8.8: Events that have been detected at the same time as the reported Earthquakes

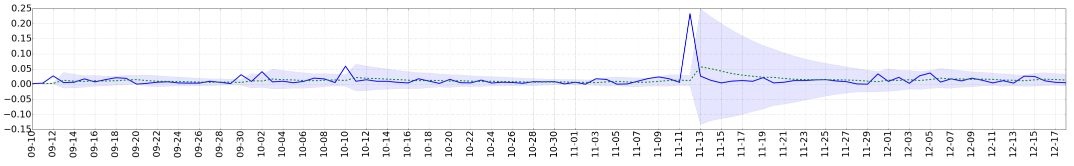
Date	σ	event terms	description
2014-09-17T02:50	247.8	#selfiesfornash, nash, #selfiefornash, #sefiesfornash	Teenager star "Nash Grier" asked his fans for selfies
2014-12-01T02:56	223.0	United_States_of_America, #soultrainawards	Soul Train Music Awards
2014-12-01T02:00	167.2	#thewalkingdead, beth, di, cry, #ripbeth, #asknorman, daryl	TV Series "The Walking Dead"
2015-01-07T22:00	135.1	hopkins, kati, England, #cbb, United_Kingdom, London, Greater_London, North_West_England	Katie Hopkins passes judgement in TV Show "Celebrity Big Brother"
2014-11-12T23:00	122.2	#followmehayes, hay	Fans send "folow me please" wishes to Teenager star "Hayes Grier"
2014-09-17T20:00	113.1	England, yaya, #gbbo, United_Kingdom, Greater_London, London, North_West_England	Famous TV Show "The Great British Bake Off" live
2014-11-20T04:05	102.3	United_States_of_America, #ahsfreakshow, New_York, Massachusetts	Fans of TV show "American Horror Story" talking about new Release Date
2014-09-17T20:51	98.0	boateng, goal	Jerome Boateng (Bayern Munich) scored the winning goal against Manchester City
2014-10-10T21:00	83.8	#5sosgoodgirls, #5sosgoodgirlsmusicvideo, 5so	Famous Teenager Band release new video
2014-10-10T10:42	70.3	kailash, satyarthi, Nobel_Peace_Prize, malala, Malala_Yousafzai	Malala and Kailash Satyarthi win Nobel Peace Prize
2014-11-12T04:38	66.1	#soa, #soafx, #finalride, abel	TV Series "Sons of Anarchy"
2015-01-28T21:00	59.8	Lionel_Messi, Neymar, suarez	Famous soccer players in Atlético Madrid against Barcelona.
2015-01-28T21:53	55.3	eriksen, Greater_London, London	Christian Eriksen scored a soccer goal



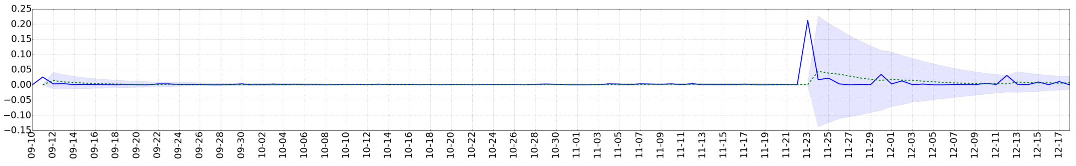
(a) Mentions of term “earthquake” regardless of location.



(b) Mentions of term “earthquake” in Guam at latitude 144 ± 1 and longitude 13 ± 1



(c) Mentions of term “earthquake” in Harper, Kansas at latitude 41 ± 1 and longitude 113 ± 1



(d) Mentions of term “earthquake” in Dallas, Texas at latitude -97 ± 1 and longitude 33 ± 1

Figure 8.7: Frequency of the term “earthquake” globally vs. locally. The dashed line thereby correspond to the average term frequency $\overline{EWMA}$, whereas the background fill shows the moving Standard Deviation $\sqrt{EWMVar}$.

Many earthquakes would have been detected by our Event Detection algorithm without using geo locations, since people and bots mention their location in the text: only the 1st (Oklahoma) and 9th (Texas) earthquake would have been missed otherwise. By including the geo information and tracking statistics for the term-location pairs (*earthquake*, *Oklahoma*) respectively (*earthquake*, *Texas*) they were identified. As shown in Table 8.10, we are able to detect 9 out of 11 earthquakes that were classified as significant by the USGS and present in our data set. GeoScope only detected a single *#earthquake* hashtag at January 7, 2015. The columns *Time* and *M* report the USGS reported earthquake date and magnitude. Δ contains the delay in minutes from the

Table 8.9: Events Validated via WikiTimes [184]

Date	σ	Event Terms	Event description © Wikipedia, The Free Encyclopedia
09-17	11.4	college, North_West_England, England, city, UK	At least 13 people are dead and 34 injured after Boko Haram gunmen attack a government college in the northern Nigerian city of Kano.
09-17	7.5	Scotland, U_K, poll, England, referendum, result, people	Voters in Scotland go to the polls to vote on the referendum on independence.
09-17	5.6	house, rebel, arm, Syria, approv	The United States Senate passes a budget measure authorizing President Barack Obama to equip and train moderate rebels to fight ISIL in Syria.
09-18	12.4	England, Scotland, referendum, U_K, Greater_London, London	Voter turnout in the referendum hits 84.5%, a record high for any election held in the United Kingdom since the introduction of universal suffrage.
09-18	25.6	Scotland, U_K, uk, England, Greater_London, London, David_Cameron	Prime Minister David Cameron announces plans to devolve further powers to Scotland, as well as the UK's other constituent countries.
09-18	15.0	England, referendum, Greater_London, U_K, Alex_Salmond, Scotland, resign, London, salmond, Glasgow_City	Alex Salmond announces his resignation as First Minister of Scotland and leader of the Scottish National Party following the referendum.
09-18	9.5	Philippines, cancel, flood, Metro_Manila, UK	Manila is inundated by massive flooding causing flights to the international airport to be cancelled and businesses to shut down.
09-22	40.1	Isis, U_S_A, Syria, airstrikes, bomb, target, islamic_state, u_s, strike, air	The United States and its allies commence air strikes against Islamic State in Syria with reports of at least 120 deaths.

Continued on next page

Table 8.9 – continued from previous page

Date	σ	Event Terms	Event description © Wikipedia, The Free Encyclopedia
09-23	17.7	Syria, strike, air, Isis	The al-Nusra Front claims its leader Abu Yousef al-Turki was killed in air strikes.
09-24	3.1	Algeria, french, behead	Algerian jihadist group Jund al-Khilafah release a video showing French tourist Hervé Gourdel being killed.
09-26	3.5	Iraq, Isis, air, strike	The Parliament of the United Kingdom approves air strikes against ISIS in Iraq by 524 votes to 43.
10-04	6.2	England, city, UK, North_West_England, Texas	The Kurdish city of Kobane in the Raqqa governate is on the brink of catastrophe as ISIS fighters besiege the city.
10-06	3.1	people, U_S_A, California, Los_Angeles_County, dead	At least four people are dead as Typhoon Phanfone makes landfall in central Japan.
10-08	60.5	di, patient, thoma, duncan, eric, dallas, hospital, diagnos, texas	The first person who was diagnosed with Ebola in the United States, Thomas Eric Duncan, a Liberian man, dies in Dallas, Texas.
10-10	44.7	kailash, satyarthi, India, Nobel_Peace_Prize, malala, Malala_Yousafzai, congratul, #nobelpeaceprize, indian, pakistani, peace	Pakistani child education activist Malala Yousafzai and Indian children's rights advocate Kailash Satyarthi share the 2014 Nobel Peace Prize.
10-12	11.0	texas, worker, posit, tests, health_care, supplement	A Texas nurse tests positive for Ebola. The health care worker is the first person to contract the disease in the United States of America.
10-14	34.4	Republic_of_Ireland, ireland, U_K, Germany, England, John_O'Shea, Leinster, County_Dublin, Scotland	Ireland stuns world champion Germany in Gelsenkirchen, with Ireland drawing the match at 1–1 when John O'Shea scores in stoppage time.

Continued on next page

Table 8.9 – continued from previous page

Date	σ	Event Terms	Event description © Wikipedia, The Free Encyclopedia
10-14	30.6	Albania, Serbia, U_K, England, London, match, drone, flag	The game between Albania and Serbia is abandoned after a drone carrying a flag promoting the concept of Greater Albania descends onto the pitch in Belgrade, sparking riots, mass brawling and an explosion.
10-15	17.8	posit, worker, tests, Ebola_virus_disease, texas, health	A second health worker tests positive for the Ebola virus in Dallas, Texas.
10-17	13.1	czar, Barack_Obama, klain, ron, Ebola_virus_disease, U_S_A, Travis_County	Barack Obama names lawyer and former political operative Ron Klain as "ebola czar" to coordinate US response to the Ebola outbreak.
10-22	26.4	soldier, Canada, Ottawa, shoot, Ontario, insid, canada, parliament	A gunman shoots a Canadian Forces soldier outside the Canadian National War Memorial.

Table 8.10: Significant Earthquakes that exhibit a minimum of 10 tweets within 1 hour of the actual time.

Time	Lat	Lng	Nearby City (distance)	M	Δ	σ	$\frac{\lvert T_{t,q,g}\rvert}{\lvert T_{t,q}\rvert}$	Event Keywords	[41]	Our
14-09-17	144.4	13.8	Hagatna, Guam (47km)	6.7	2.3	21.7	93/114	guam, {guam, earthquake}, {guam_county, earthquake}	✗	✓
14-10-02	-98.0	37.2	Wichita, KA (73km)	4.3	-	-	13/51	-	✗	✗
14-10-10	-96.8	35.9	Oklahoma, OK (86km)	4.2	53.7	3.0	41/54	oklahoma, {oklahoma, earthquake}	✗	✓
14-11-12	-97.6	37.3	Wichita, KA (53km)	4.9	2.0	18.0	138/322	kansas, {earthquake, felt}, {kansas, earthquake}	✗	✓
14-11-20	-121.5	36.8	Santa Cruz, CA (65km)	4.2	0.2	18.7	134/148	california, {california, earthquake}, {monterey_county, earthquake}	✗	✓
14-11-21	127.1	2.3	Bitung, Indon. (228km)	6.5	-	-	13/19	-	✗	✗
14-12-01	-111.7	35.0	Phoenix, AZ (179km)	4.7	3.4	22.8	119/148	arizona, {arizona, earthquake}, {earthquake, flagstaff}	✗	✓
14-12-30	-118.3	33.6	Long Beach, CA (21km)	3.9	0.7	22.5	213/245	california, {california, earthquake}, {los_angeles, earthquake}	✗	✓
15-01-07	-96.9	32.8	Irving, TX (6km)	3.6	22.9	3.1	269/334	{denton_county, earthquake}	✓	✓
15-01-20	-121.0	36.4	Santa Cruz, CA (80km)	4.4	3.4	8.0	37/51	california, {california, earthquake}, {monterey_county, earthquake}	✗	✓
15-01-28	-124.6	40.3	Sacramento, CA (330km)	5.7	51.1	3.0	25/37	earthquake	✗	✓

Note: data and method were *not* optimized for earthquake detection, but *all* events were tracked at the same time. The detected earthquakes were significant at a significance level of $\sigma \geq 3$ in the full Twitter feed.

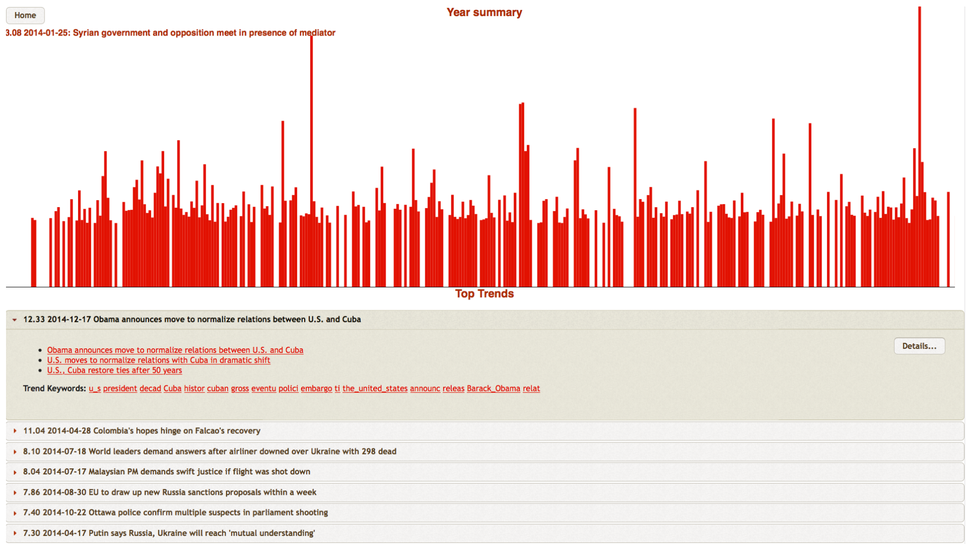
earthquake to our first detected event that surpassed the threshold of 3σ. $T_{t,q}$ denotes the set of all tweets within a 1-hour window of the event matching the query term $q =$ *"earthquake"*. $T_{t,q,g} \subseteq T_{t,q}$ denotes the subset of tweets, which also match our geo token g of the earthquake's location. *Event Keywords* are the terms and pairs that were significant. In the last columns we indicate whether *GeoScope* and *our* method were able to identify this event.

We did *not* optimize our system specifically for this purpose, in contrast to specialized systems such as Sakaki et al. [157], by specifying query terms beforehand. We first detected all events, and then filtered the earthquake related events from the log file for evaluation. To emphasise this fact, we extracted all reported trends which occurred at the same day as these earthquakes. For a more meaningful presentation we clustered all detected event pairs by an hierarchical agglomerative clustering approach with average linking strategy to aggregate co-occurrence. The results can be reviewed in Table 8.8.

As this experiment demonstrates that our method is able to detect real world events, it also reveals the weaknesses of the overall idea of an earthquake warning with Twitter data: as discussed above, only 19 out of 30 significant earthquakes were not significantly discussed on Twitter. But on the other hand, most happen off-shore, such as the last (Sacramento) earthquake in this list, which happened 40 miles off-shore south-west of Eureka, CA and was mostly reported by Twitter bots and do not cause much Twitter activity. Furthermore, when a substantial earthquake happens, we must assume the network to become unavailable. While this is a popular use case for event detection on Twitter, it cannot be used as a replacement for seismic sensors.

8.8 Online Demonstration

The results of our trend detection system are available at `http://signi-trend.appspot.com/` for exploration. An example screenshot can be examined in Figure 8.8 and Figure 8.9.



3.08 2014-01-25: Syrian government and opposition meet in presence of mediator

12.33 2014-12-17 Obama announces move to normalize relations between U.S. and Cuba

- Obama announces move to normalize relations between U.S. and Cuba
- U.S. moves to normalize relations with Cuba in dramatic shift
- U.S., Cuba restore ties after 50 years

Trend Keywords: u_s president decad Cuba histor cuban gross eventu polici embargo ti the_united_states announc releas Barack_Obama relat

11.04 2014-04-28 Colombia's hopes hinge on Falcao's recovery

8.10 2014-07-18 World leaders demand answers after airliner downed over Ukraine with 298 dead

8.04 2014-07-17 Malaysian PM demands swift justice if flight was shot down

7.86 2014-08-30 EU to draw up new Russia sanctions proposals within a week

7.40 2014-10-22 Ottawa police confirm multiple suspects in parliament shooting

7.30 2014-04-17 Putin says Russia, Ukraine will reach 'mutual understanding'

Figure 8.8: Web app overview of year summary

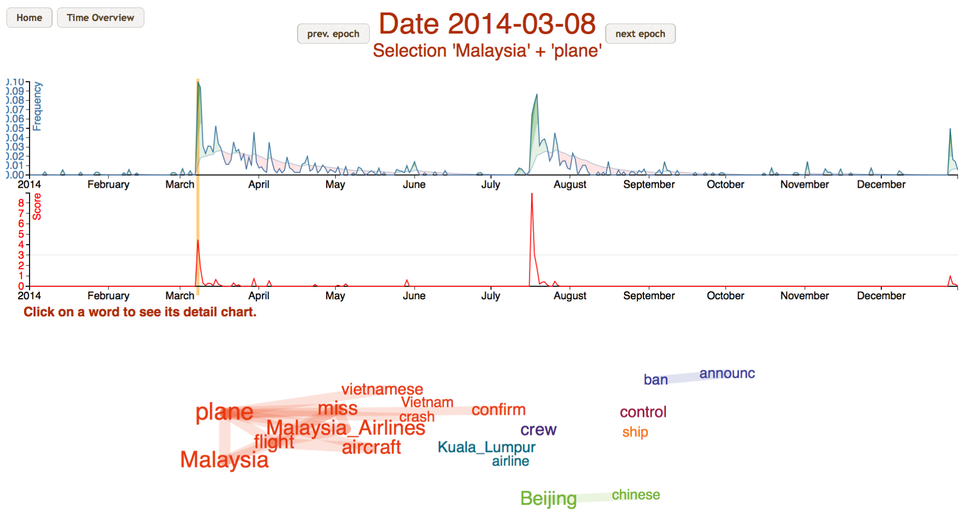


Figure 8.9: Web app detail for the Malaysia airplane crash

Chapter 9

Relationship of Trends and Outliers

As previously discussed in Part II, trends have a strong relationship to outliers as they are some kind of unusual observation in our data set. The following chapter is intended as a call to action. Outlier detection research has been very much focus on Euclidean space, and the community has become detached from the actual data problems we want to solve. This manifests itself in a lack of good evaluation data, and in often incremental variations of the general theme [44]. *Data diversity* is a leading theme of data science: much of our data at hand cannot be squeezed into the rigid structure of a finite $\mathbb{R}^d$ vector space. If we want to obtain meaningful *descriptions* of outliers, we first need to work on meaningful *data*. This can be achieved by working closer to the original data, and less on an abstract vectorized representation. For this, flexible and abstract methods are needed that can be adapted to the different actual problems to solve.

Outlier detection, while meant to be an *unsupervised* task, is not—and cannot be—entirely free from assumptions on the characteristics of outliers. Every existing method embodies some implicit concept of outlierness. And even if methods do not use dedicated training data, they are still "trained" by the intuition of the method designer of what constitutes normal or abnormal.

We suggest to formalize this notion, make it explicit, and design methods that allow customized notions of outlierness. Eventually, this will also lead to better explanation and description of outliers.

In this chapter, we focused on the peculiarities of textual data, which often comes streaming and in a high volume. We interpret existing trend detection methods as simple outlier detectors over time series. Many of the abstract problems become visible in this context, such as the difference between instances (e.g., messages) and outliers (i.e., trends or events) in this domain. Reflecting traditional Outlier Detection, we observe a similar pattern in density-based Outlier Detection: as much as we are seeking outlier instances, we are also seeking regions of low density.

In summary, we suggest the following *research issues*:

- Improved statistical models need to be developed and used, to obtain meaningful conclusions and provide robustness against spam.
- Low latency is required for practical use for trend detection, which collides with the desire to use larger time windows to obtain more reliable statistics. This may be resolved by using an appropriate combination of models [207].
- Aggregation of results is important, e.g., merging overlapping trending topics, to avoid overloading the user with noisy and redundant results.[1]
- Scalability to a large number of instances, to a large number of aggregations, and to a fast data stream is required. This will usually require the use of approximation and indexing techniques, and will limit the complexity of models usable [171].
- Artifacts and domain-specific anomalies are omnipresent in real data, and it should be possible to customize and modify methods to handle these. On text data, stop words and spam constitute such artifacts. On census data areas with few inhabitants cause such anomalies. In traffic data, accidents attributed to the nearest milepost may mislead an algorithm.
- Bridge the gap! Outlier models should be developed that are applicable to Euclidean vector spaces, time series, text, and other data types in the same way. This will make it easier for one research domain to benefit from advances in the other.

9.1 Introduction

As introduced in Section 3.7, outliers are hard to define mathematically because we cannot expect them to follow a model or distribution known beforehand. In this chapter, we want to expand the generalized framework for Outlier Detection proposed by Schubert et al. [170] to also cover event detection and emerging topic detection. Outlier detection is commonly defined as the process of finding unusual, rare observations in a large data set, without prior knowledge of which objects to look for. Trend detection is the task of finding some unexpected change in some quantity, such as the occurrence of certain topics in a textual data stream. Here, the task is to detect changes in the distribution of a data stream that indicate the beginning of an event. The words used are slightly different, e.g. trend, emerging topic, bursty keyword. But essentially

[1]In many areas of data mining, redundancy of data mining results has been observed for early approaches and has been addressed later on in more mature approaches, well-known examples being subspace clustering [174] and frequent pattern mining [211].

they refer to unusual, extreme topics: outliers in text streams. Many established Outlier Detection methods are designed to search for low-density objects in a static data set of vectors in Euclidean space. For trend detection, high volume events are of interest and the data set is constantly changing. These two problems appear to be very different at first. However, they also have obvious similarities. For example, trends and outliers likewise are supposed to be rare occurrences. In this chapter, we discuss the close relationship of these tasks. We call to action to investigate this further, to carry over insights, ideas, and algorithms from one domain to the other.

The remainder of this chapter is organized as follows. In Section 9.2, we survey traditional Outlier Detection methods (designed for Euclidean spaces) that are abstract in the sense that they are designed for Euclidean data spaces. Although there are specializations to particular data types, in many cases the wheel is reinvented for such particular data types. A recent discussion of generalized approaches eventually allows for transfer of insights from traditional methods to specialized applications like trend detection.

In Section 9.3, we point out why the research questions guiding the design of Outlier Detection methods might be misleading in some cases and why traditional Outlier Detection might also benefit from insights in trend detection. In Section 9.4, we discuss some methods for trend detection and their relationship to a generalized view of traditional Outlier Detection. Finally, in Section 11.6, we summarize and identify challenges for future research in these areas.

9.2 Traditional Outlier Detection

9.2.1 Outlier Detection in Euclidean Space

Knorr and Ng [94] proposed a distance-based notion of outliers. This model is motivated by the intuition of statistical parametric approaches. It is aiming, however, not on a refinement of the statistical modelling of outliers but at designing efficient database-oriented approaches. This algorithm triggered the data mining community to develop many different approaches that have a less statistically oriented but a more spatially oriented notion to model outliers. The k-NN-outlier model [151] ranks the objects according to their distances to their k-th nearest neighbor. As a variant, the k-NN-weight model [24] uses the sum of distances to all objects within the set of k nearest neighbors (called the weight) as an outlier degree. While these models actually only use distances, the intuition is typically discussed with Euclidean data space in mind. Many approaches were primarily interested in algorithmically improving efficiency, for example based on approximations or improved pruning techniques for mining the top-n outliers [33, 95, 186, 22]. Several efficient or approximate algorithms for mining distance-based outliers have been studied by [145]. They identify common algorithmic techniques but do not discuss model properties, restricting themselves to the distance-based models [94, 151, 24].

In these distance-based approaches, for each object, a property (outlier model) [170] is learned based on a local neighborhood (radius ε, k nearest neighbors). However, the objects are eventually ranked according to this property ("outlier score") in a global way. For example, the object with the largest k-NN distance overall would be the most prominent outlier. Thus, these methods are best suited to identify *global* outliers. Recent global approaches base the decision not on Euclidean distances but on angle-variance (such as ABOD [104] and an efficient variant using random projections [147]), an intuition that is clearly also connected to a Euclidean data space.

Identifying *local* outliers (i.e., comparing local models with a local reference set [170]) started with the method LOF (local outlier factor) [39] as introduced in Section 3.7. Again, this notion of outlierness is a natural intuition for the Euclidean data space.

Several extensions and refinements of the basic LOF model have been proposed, e.g. a connectivity-based outlier factor (COF) [180], or using the concept of micro-clusters to efficiently mine the top-n density-based local outliers in large databases (i.e., those n objects having the highest LOF value) [89]. A similar algorithm, named INFLO [90], for an extension of the LOF model is using also the reverse nearest neighbors additionally to the nearest neighbors and considering a symmetric relationship between both values as a measure of outlierness. The local distance-based Outlier Detection (LDOF) approach [203] merges the notion of local outlierness with the distance-based notion of outliers. LoOP [100] uses a density estimation based on the distance distribution of all nearest neighbors and formulates the local outlier score as a probability. COP [103] aims at detecting outliers in the presence of local correlations in the data set by measuring the deviation from the local model.

More or less explicitly, all these methods basically aim at providing rather simple approximations of statistical density estimates around data points in Euclidean space. Consequently, a recent evaluation study [44], discussing several of these methods, also focuses on numeric data.

9.2.2 Specialized Outlier Detection

Some approaches designed for high-dimensional data try to account for a local feature relevance and search outliers in subspaces of the data space [138, 101, 139, 140, 143, 91, 137, 135, 60], see the survey of Zimek et al. [210]. In the area of spatial data mining [154], the topic of spatial outliers has triggered several specialized methods [25, 173, 126, 96, 176, 50, 125, 52]. These approaches discern between spatial attributes (relevant for defining a neighborhood) and other attributes (usually only one additional attribute) where outliers deviate considerably from the corresponding attribute value of their spatial neighbors. How to derive spatial neighborhood and how to define "considerable deviation", however, differs from approach to approach. Other specialized approaches tackle for example outliers in time series [179, 86], outliers in graphs (e.g., in social networks or in DBLP) [70, 16, 15, 17], outlying trajectories [114], outliers in categorical or ordinal data [18, 202], or in uncertain data [13]. Aggarwal [11] provides more examples. Again, although they are not always operating in Euclidean space, all these methods aim eventually at some approximate descriptors of outlierness for the objects that ultimately should relate to statistical density estimates.

9.2.3 Outlier Detection in Data Streams

Recently, Sadik and Gruenwald [156] gave an overview on research issues for Outlier Detection in data streams. They follow the categorization of the well-known survey on Outlier Detection by Chandola et al. [48], where type II outliers, as opposed to type I outliers, are outliers with respect to some context, such as time or location. In database research, the concept of outliers w.r.t. some particular context is also known as a "local" outlier [39, 170] and is not restricted to special data types, although the concept of "locality" might be of paramount interest in such data types with special requirements for the outlier model [170, 169]. Consequently, Sadik and Gruenwald [156] are interested in streaming data and time series data without distinction, while we see time series as a special data type but consider here the scenario of streaming data as a more general scenario, that has been tackled in many studies with Euclidean data space in mind [199, 175, 149, 23, 28, 99, 72, 105]. There are two main categories of tackling dynamic data. First, the dynamic aspect of the data is tackled using an incremental approach, i.e., old data remain available while new data are coming in and the preliminary models are refined over time. The second possibility is to truly address the aspect of potential infinity of data, i.e., the fact that the complete data stream might not fit into the available memory or might actually never be completely available. In this case, the typical approach is a sliding "time window" that is oblivious of old data. The adapted approach therefore builds a model based only on the data within the time frame of the window.

9.2.4 Generalization of Outlier Detection

In a certain sense, the combination of different outlier detectors into an ensemble [113, 71, 142, 102, 168, 124, 209, 207, 208] can be seen as a generalization because, under certain conditions [102, 168], it becomes meaningful to combine even different methods that follow different intuitions about outlierness. But still, such combinations can only combine the available methods, that have been typically designed for Euclidean data space or for some particular use case. There is a recent line of reasoning, though, on truly generalizing the classic, abstract outlier methods to new use cases and data scenarios. Schubert et al. [170] modularized many existing Outlier Detection methods, demonstrating that there is a large conceptual overlap in these methods. Based on this modularized structure, they demonstrate how to modify existing methods to work on other data types such as geostatistical data, video streams, and graph data.

In follow-up work [169], they introduced and demonstrated improved density estimation modules with adaptations to spatial-temporal data. These generalization studies constitute important prior work to what we will discuss next, but they focused on identifying common patterns and applying the discussed methods to other data types. Here, we will discuss a further generalization which also integrates data preprocessing aspects into this broader view, as this generalized pattern cannot easily encompass trend detection, despite the strong similarities between these two domains.

9.3 Limitations of Traditional Outlier Detection

Most of the published Outlier Detection methods were designed with the intuition of low-density outliers in mind. While this is applicable in some scenarios, it often does not fit a given problem. The difficulties of putting Outlier Detection methods from theory into practical use (or even generating test data sets [66]) may be attributed to overusing this intuition.

In the following examples, we want to discuss some scenarios where data do not adhere to the intuition of low-density, and traditional Outlier Detection methods then do not work reliably. As a consequence, they would not produce satisfactory results. We will also discuss what should have been analyzed instead.

9.3.1 Example: KDD Cup '99

On the popular KDD Cup '99 data set, one may argue that outliers are not at all rare instances. Depending on the exact version of this data set, $80\% - 94\%$ of the instances are attacks. As such, the legitimate connections may be considered the anomalies here. While this data set has been repeatedly used for evaluating Outlier Detection methods [112, 199, 113, 5, 200, 142, 102, 168, 124], (usually by considering only attacks of types `U2R` and `R2L` as anomalies) the results of such analyses should be taken with a grain of salt.[2] The data set has many ($\approx 75 - 78\%$ [181]) duplicates and many established methods are not prepared for handling too many duplicates. Thus, unless the outlier methods are carefully implemented and parametrized, outlier scores may become undefined, and evaluation may be biased. For example, methods such as isolation forests [124] that work on random samples of the data set might then appear to perform better because they are less susceptible to the problem of duplicates.

For the NSL-KDD version of this data set [181], attempts have been made to alleviate some of the deficiencies. Nevertheless, the attacks do not resemble modern network traffic or attacks—for example, common modern attacks such as SQL injections are not captured by this data and it contains 50% malicious

[2]See the discussion by McHugh [133] and by Tavallaee et al. [181] and kdnuggets n18 2007, "KDD Cup '99 dataset (Network Intrusion) considered harmful": `http://www.kdnuggets.com/news/2007/n18/4i.html`

activity, almost completely of bulk type— and the data set is not suitable for evaluating intrusion detection capabilities. Because it originates from a simulated network, we may however treat this as a synthetic test set for benchmark purposes.

Because the data set contains categorical attributes, binary attributes, and integer valued attributes (including e.g., `num_compromised`), it is highly sensitive to preprocessing such as feature selection and data normalization. Furthermore, any intuition of density and distance, based on Euclidean space, is probably inappropriate for this data set.

9.3.2 Example: United States Census Data

In spatial Outlier Detection [7], observations consist of two kinds of data: a geographical location—which may be a point (a position) or an area (a polygon)—as well as a univariate or multivariate measurement. The US census data for example, include statistics such as household size and population demographics at different spatial resolution such as census counting districts and county level. [3] For some districts, sparsity of population causes artifacts: census counting districts include areas such as airports, graveyards, and ghost towns with a low population. Popular attributes such as relative ethnicity may be undefined for uninhabited districts or show unusually extreme values for tiny populations. Therefore, popular Outlier Detection algorithms such as LOF cannot be meaningfully used on such data without modifications [170]. However, the methods can be easily generalized in a way to use a spatial context to determine the neighborhood and the non-spatial attributes for analysis and yield results competitive to those of existing geostatistics [170]. To make full use of this data set, the methods should be further customized to take uncertainty into account, in particularly those arising from a small population, which makes numbers such as ethnicity averages incomparable.

9.3.3 Example: Traffic Accidents

Schubert et al. [169] analyzed the density of traffic accidents in the UK, based on open government data.[4] Again, results obtained by traditional Outlier Detection methods are not helpful: they will report accidents in low populated areas such as northern Scotland as low-density outliers. The data set contains 19% duplicated coordinates, probably due to measurement precision and reoccurring accident sites. Information available may include involvement of pedestrians, visibility conditions, severity, casualties, road numbers, authority IDs etc. that may also be missing or estimated. For their analysis, they only used the coordinates, and customized their approach for this data set by searching for areas with higher traffic accident density than expected, in order to find accident hotspots.

[3]Available at the United States Census Bureau `http://www.census.gov/`

[4]Publicly available on `http://data.gov.uk/`

9.3.4 Observations

Above examples demonstrate how we might have been *asking the wrong question* in Outlier Detection research. By working primarily with data consisting of vectors in Euclidean space, and the intuition of low-density outliers, we designed our algorithms for this particular use case. This can be seen as a kind of "overfitting" at algorithm design time. Similar observations have been made in other fields of data mining, e.g., clustering [69] and pattern mining [212].

On the other hand, we have also been using our existing Outlier Detection tools the wrong way. Both on KDD Cup '99 and on the traffic accidents example our entities of interest are *not* the individual data samples. Instead, our potential outliers are some aggregation of the data: we need to aggregate the KDD Cup '99 data to *hosts* instead of processing individual connections if we want to detect attackers; Schubert et al. [169] implicitly used local maxima in density as aggregations of traffic accidents (which yields black spot crossroads, not "outlier car accidents"). On the U.S. Census data, the data were already aggregated by the Census Bureau. When analyzing trends in text, we are interested in *topics, not messages.* This also holds for first story detection (FSD), where the output is not the topic, but the first message of each topic. This is related to what has been termed "type III outliers" [48] with the distinction that we claim that the particular type of aggregation is typically not a priori *given* with the data but is a matter of an adequate *interpretation* of the plain data. One may argue that this can be solved by improving feature extraction or preprocessing (e.g. converting textual data to numerical vectors, aggregated at the desired level) before analyzing with an Outlier Detection method. Likewise one may argue that this could be done in the preprocessing phase. Simply convert the textual data to appropriate numerical vectors, aggregate and transform the numeric data to the desired level (i.e. topics, instead of messages), and then analyze the aggregated and transformed data with some (Euclidean space) Outlier Detection method. In practice however, as we will see next, this does not work that easily. On data streams, preprocessing and transformation cannot be completely decoupled when to much assumptions are made in advance. If we want to be able to explain the resulting outliers, we need to be able to return to the original data representation. Last but not least, because of efficiency considerations, it may be necessary to integrate outlier/trend detection much earlier in the analysis process, instead of first transforming all data objects. For this reason we designed our Event Detection discussed earlier to be able to work on *all* input data without selecting only a fraction of the data (such as hashtags).

9.3.5 Bridge the Gap

If we want to bridge the gap between Outlier Detection research and trend detection (as well as other domains), we need to go beyond the idea that a continuous $\mathbb{R}^d$ vector space without duplicates, where low-density regions con-

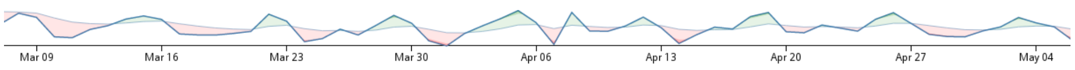


Figure 9.1: “Pizza” seems to be particularly popular on weekends. Screenshot from `http://signi-trend.appspot.com/`

tain the outliers, is a useful representation. We should bring ideas from Outlier Detection to the data types and representations that are used in practice.

9.4 Relationship between Trends and Outliers

By looking at all these studies discussed in the previous sections about Event Detection, a common important component is the definition of the properties (like frequency or density) with their expected *normal* range to determine outliers within the document space. Where trend detection is interested in outliers in textual streams, these outliers are not individual instances—messages, news items, tweets—but rather *topics*. If we attempt to run traditional Outlier Detection methods on such a stream, we will get plenty of uninteresting outliers due to misspellings and rare words. By a classic notion of outlierness, such instances will most correctly appear as outliers. However, in most cases we probably do not want our algorithms to degenerate to counting the number of rare words per message and find the text with the most unusual vocabulary in the corpus.

Most trend detection methods such as TwitterMonitor [132], Burst Detection [76], enBlogue [21] as well as the Event Detection algorithm proposed in this thesis, perform some kind of aggregation. Often, a sliding window approach is used for aggregation of individual instances. With this technique, each such window corresponds to a point in time, and the aggregation yields one such time series for every word (or n-gram) in the data set. Outliers are those time series that show an unusual change in activity in the current time window.

When interpreting these example algorithms for trend detection on the time series of a single term, we can roughly summarize their model as follows: TwitterMonitor uses the increase in term activity compared to the previous time window, Burst Detection uses the second derivative (the increase in the current time window, compared to the previous increase), enBlogue uses the relative increase in frequency over a moving average. Our algorithm uses the most complex model, consisting of exponentially weighted moving average and standard deviation, and it thus can also capture variance. All of these are fairly simple statistical models and in itself not spectacular. Much of the challenge in trend detection comes from scale: these values must be tracked and analyzed for every word (or n-gram) in the data set simultaneously, for millions of words. The main contributions of above articles are on scalability, not the statistical models used: enBlogue tracks only those word pairs where at least one word is considered a seed tag. We are using a hashing-based approach, which is lossy

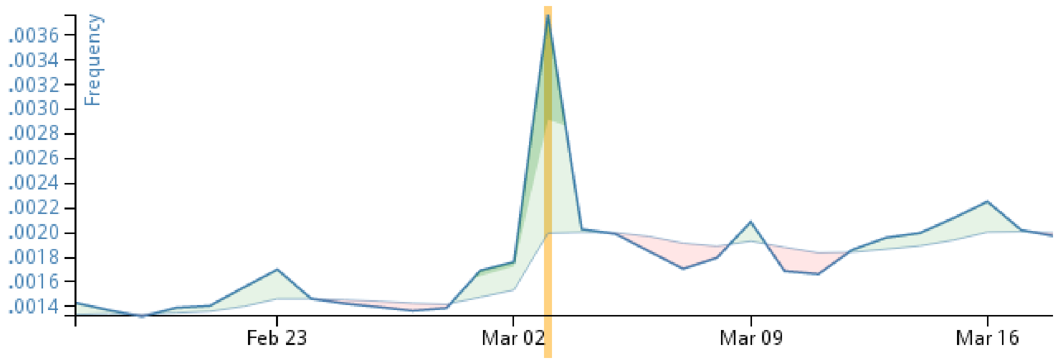


Figure 9.2: Statistical model used by SigniTrend: frequency (thick line), mean (thin line) and standard deviations (shaded areas) for the keyword "Selfie" on Twitter. The yellow bar highlights the outlier event. Screenshot from `http://signi-trend.appspot.com/`

on rare terms but accurate with high probability on frequent terms and thus can afford using only a constant amount of memory. All these methods can be seen as first-order Outlier Detection methods on time series. The earlier frequencies of a term are used to construct a model (frequency, moving average, moving average with standard deviation) and the new frequency is compared to the previous value. Figure 9.2 recalls Event Detection model: from the observed frequency a moving average as well as a moving standard deviation are computed. The famous "Oscar Selfie" achieved 11 standard deviations over the previous average. We can also see that, after this selfie, the average volume substantially increased (note that the axis does not start at 0), but the model adapted quickly to this higher average volume. If we consider the standard deviation as used by our method to be a derived value from the mean, then this method can be seen as second-order. We have not yet, however, observed "locality" [170] in the sense of LOF [39]: each time series object is evaluated on its own, the significance is not compared to other time series.

The time series of "pizza" (see Figure 9.1) shows a typical weekend pattern, except for Oscar night. Comparing this series with the series of related terms may make such outliers clearer to detect. This calls for future work, as this will eventually allow to detect trends in smaller communities, that otherwise are masked by globally popular trends (the so called "Justin Bieber effect").

On the other hand, several related terms (e.g. *boston*, *marathon*, *explosion*) may trend together, and the comparison of the series may help identifying the most explaining term combination. Judging the significance of a trend based on the scores of related terms can thus be expected to yield better results. This way, lessons learned for traditional Outlier Detection, reconsidering the notion of "locality" [170], could be transferred and boost research progress also in trend or event detection.

Chapter 10

Conclusion

In Part II of this thesis we showed how the use of exponentially weighted floating averages and variance to score a trending topic with respect to its recent occurrences in the data stream. The proposed statistic needs little memory (two floating point values only) and can be efficiently updated using an incremental equation. The numerical properties of this equation are well understood, and it was shown to be more stable than the naive equation of the variance involving a difference of squares. This improved scoring function is able to capture the background noise as well as general trends in the data, and by measuring variability it can produce a much more meaningful significance score than e.g. nearest-neighbor distances that were used in state-of-the-art prior work to measure emerging topics.

Secondly, we showed how to scale our statistical approach to track every word and word pair of a data stream with limited memory, by using hashing techniques and exploiting that the majority of words and word pairs occurs only rarely in the stream. By resolving hash collisions in favor of the more common word or word pair, we will usually have an exact statistic for these words and only occur information loss on rarely seen words – which then by definition are not trending yet. Scalability of this approach was demonstrated by analyzing state-of-the-art data sets faster than real-time on a single CPU.

Third, we suggested and demonstrated the use of clustering techniques to aggregate the observed trends – which usually only affect a small subset of the vocabulary, for which the results can be refined from an inverted index – into larger trends. This has become more important than in previous work, as we monitor a much larger set of candidates; and in particular co-occurrences tend to overlap and form clusters.

We then showed how we can extend our Event Detection algorithm with the ability to use geographic information to detect events in fast and very large data streams, and showed that:

1. Mapping coordinates to a token representation allows efficient integration of geographic information in an analysis pipeline for textual data.

2. Co-occurrence of terms and location yields insight into events happening

around the globe.

3. Variance-based normalization using incremental statistics yields more interesting topics and events by adjusting for differences in user density and activity and removes the need to expire old data.

4. Decoupling data aggregation and statistics tracking permits the use of short timeframes for counting and alerting, while at the same time we can use larger timeframes for more robust statistics.

5. Probabilistic counting using parallelization-friendly hash tables allows to scale this approach to very large data sets, well capable of performing such an analysis in real-time on thousands of tweets per minute and enables us to track any word-location combination.

6. Due to the scalability improvements, the a priori specification of interesting topics and regions of interest required by earlier approaches is no longer necessary.

To the best of our knowledge, we developed the first system that can monitor all single words, word-word pairs or word-location pairs to build a co-occurrence model on a large data stream without the need for parallelization, in comparison to previous work that can only monitor a filtered set of terms or word pairs based on a seed set. Such a restriction is no longer necessary with the memory reduction obtained via hashing and our effective incremental moving averages that makes sliding window management unnecessary and gains additional performance.

We further showed how trends and events relate to classic Outlier Detection. Thus, we interpret existing trend detection methods as simple outlier detectors over time series. Many of the abstract problems become visible in this context, such as the difference between instances (e.g., messages) and outliers (i.e., trends or events) in this domain. Reflecting traditional Outlier Detection, we observe a similar pattern in density-based Outlier Detection: as much as we are seeking outlier instances, we are also seeking regions of low density. In summary, we suggest the following *research issues*:

1. Improved statistical models need to be developed and used, to obtain meaningful conclusions and provide robustness against spam.

2. Low latency is required for practical use for trend detection, which collides with the desire to use larger time windows to obtain more reliable statistics. This may be resolved by using an appropriate combination of models [207].

3. Aggregation of results is important, e.g., merging overlapping trending topics, to avoid overloading the user with noisy and redundant results.[1]

[1]In many areas of data mining, redundancy of data mining results has been observed for early approaches and has been addressed later on in more mature approaches, well-known examples being subspace clustering [174] and frequent pattern mining [211].

4. Scalability to a large number of instances, to a large number of aggregations, and to a fast data stream is required. This will usually require the use of approximation and indexing techniques, and will limit the complexity of models usable [171].

5. Artifacts and domain-specific anomalies are omnipresent in real data and it should be possible to customize and modify methods to handle these. On text data, stop words and spam constitute such artifacts. On census data areas with few inhabitants cause such anomalies. In traffic data, accidents attributed to the nearest milepost may mislead an algorithm.

6. Bridge the gap! Outlier models should be developed that are applicable to Euclidean vector spaces, time series, text, and other data types in the same way. This will make it easier for one research domain to benefit from advances in the other.

Part III

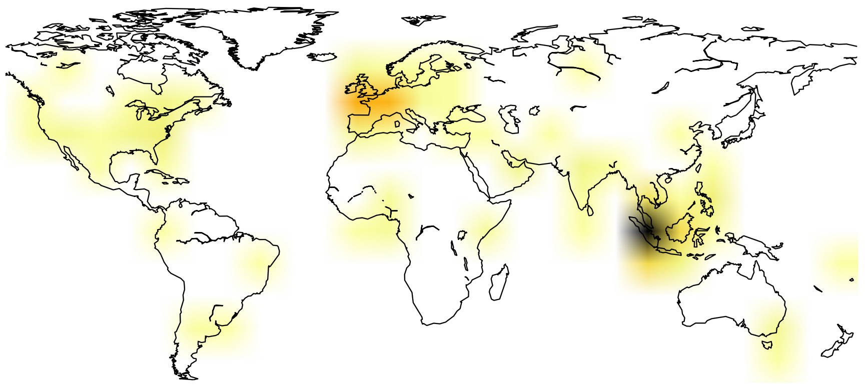
Geo-social Co-location Mining of Events

Chapter 11

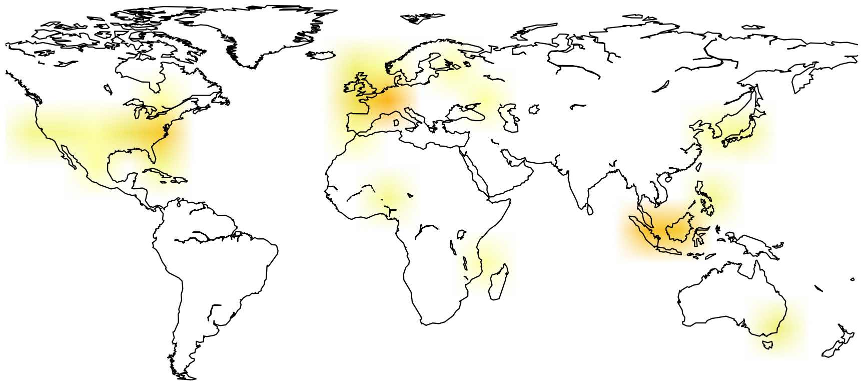
Trend Patterns and Dissimilation Prediction

The work in this chapter gives an example use case of an algorithm that benefits from a preprocessing pipeline with earlier stage of an event detection. We also discussed in Part II, how location data can be used to improve the detection performance of events. In this chapter we outline an additional usage of the location data from the remaining event candidate contained filtered from a social media data stream. This geo-textual data allows to immediately detect and react to new and emerging trends and events. As previously discussed we use the term event and trend interchangeably. As this chapter focuses on the dissimilation pattern of events over larger time periods, we will instead use the term trend to describe the observed topics. A trend is represented as a set of keywords associated with a time interval where the frequency of these keywords is increased significantly. In the following chapter, we investigate the dissemination of trends over space and time. For this purpose, we employ a four-step framework. In the first step, we employ existing solutions to mine a large number of trends. Second, for each event we create a spatio-temporal dissemination model, which describes the motion of this trend over space and time. To model this dissemination, we employ a (flow-source, flow-destination, time, trend) tensor. In the third step, we cluster these trend-tensors, to identify groups of archetypes. For each archetype, we aggregate all tensors of the same archetype, and employ a tensor factorization approach to describe this archetype by its latent features. As the fourth step, we propose an algorithm which can classify the archetypes of a new discovered trend, in order to predict its future dissemination.

In our experiments, we are able to show that the event space exhibits clusters, each corresponding to a trend-archetype such as politics, disasters and celebrity topics.



(a) July 17th 2014



(b) July 19th 2014

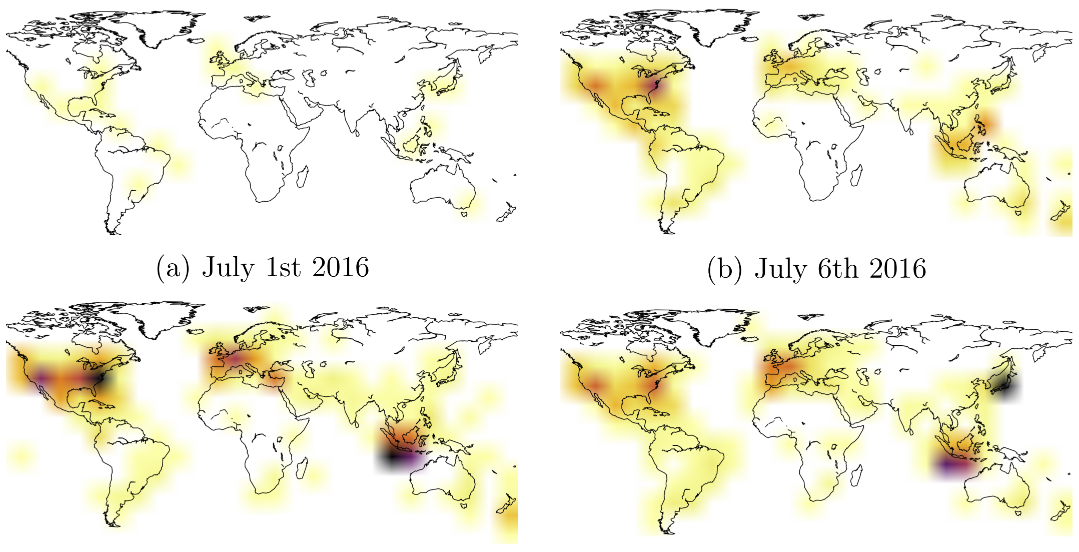
Figure 11.1: Distribution of trend "MH17"

11.1 Introduction

In this chapter, we discuss how we predict the flow of existing trends over the globe. For instance, trends related to *fashion* might often arise in France, then move over to the rest of Europe within a few days, then start to affect North America within weeks and finally flow to Australia within weeks and months. In contrast, technological trends might often be initiated in Japan and South Korea, then flow to North America, and only then flow to Europe.

As an example of such a trend behaviour, Figure 11.1 shows the location of tweets issued in July of 2014 corresponding to the lost Malaysian Airlines flight "MH17". The trend shows initial strong bursts in Malaysia as well as in the Netherlands, from where the missing flight originated, as seen in Figure 11.1a. From there, the trend quickly spread all across the world – two days later, the rest of Europe as well as North America are just as involved in the trend. This can be seen in Figure 11.1b.

A more recent trend development can be seen in Figure 11.2, where the location of tweets containing the string "Pokémon" is shown for several days. Beginning with the first of July, 2016, Figure 11.2a exhibits a globally low interest in this topic, indicating no trend at that time. As the free-to-play


(a) July 1st 2016 (b) July 6th 2016

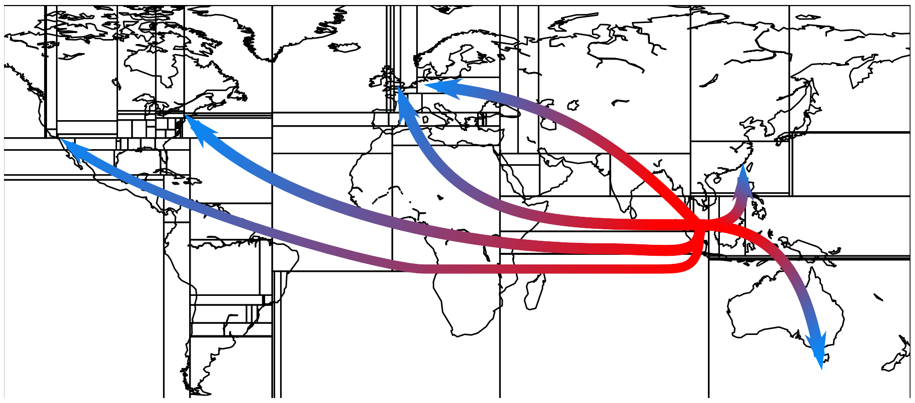
(c) July 13th 2016 (d) July 22nd 2016

Figure 11.2: Distribution of trend “PokémonGo!”

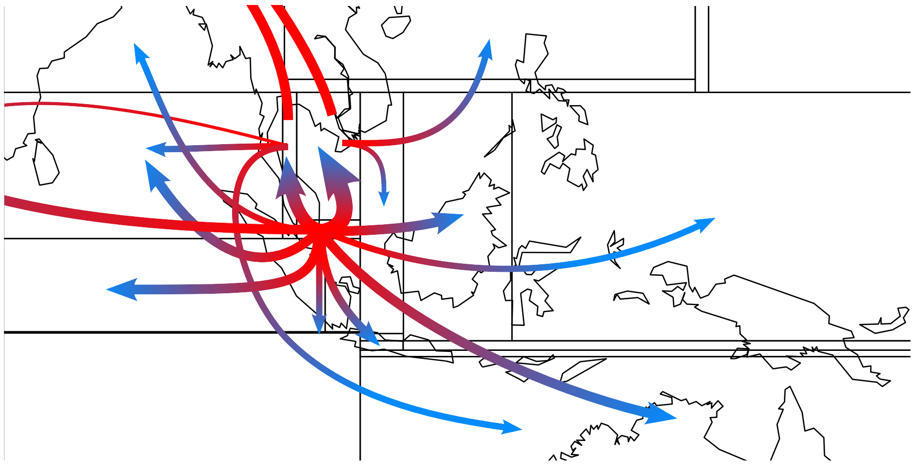
game “PokémonGo!” was released for cell phones in the United States, Figure 11.2b shows a highly significant burst of tweets on this topic on July the 6th, originating in the US alone. One week later, on July 13th, the trend has moved to Europe as the game was released in several countries there. This can be seen in Figure 11.2c. Asia follows, mainly with the Japan release on July 22nd, with a high activity regarding the topic as shown in Figure 11.2d.

Intuitively, different types of trends are expected to show different distributions. While a few trends spread to a global scale within hours due to dissemination through news networks, other trends may be more local, spread slower, might be originating from specific regions, or might disseminate to specific regions only.

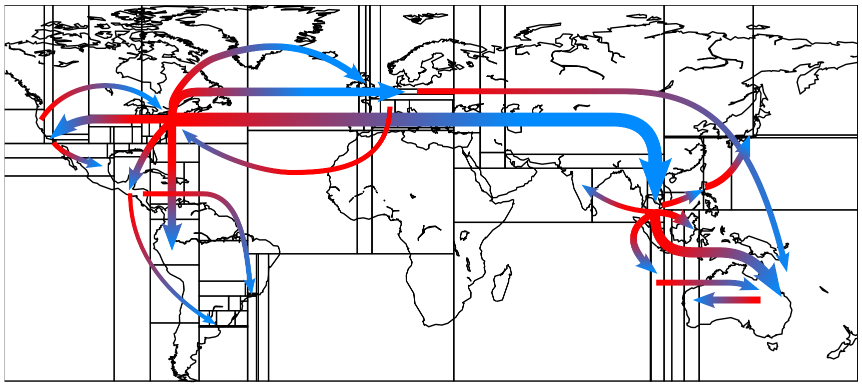
In this work we model such dissemination of trends over space and time. That is, we observe the flow of trends, specified by source and target regions, over time. Figure 11.3 exemplifies such flows for the two examples given before, namely “MH17” and “PokémonGo!”. The arrows on the map indicate a flow in activity from source (red) to target (blue). For the sake of readability, the representation has been kept coarse and omits certain regional interdependencies. Geographical regions are referenced by their position in our index (drawn in black outlines), and thickness of arrows indicates strength of the dependence. Figures 11.3a and 11.3b exhibit trend dissemination of the trend “MH17” in a full world view and one of the south-east Asian region alone, respectively. As you can see, the trend originates from Malaysia and spreads over the world from there, partially using other regions as intermediate hops. In contrast, Figure 11.3c uses the same representation for the trend “PokémonGo!” on a world-wide scale and while there is a general main direction from the US east coast, several rules in the opposite direction indicate a more diverse dissemi-



(a) “MH17” - world map



(b) “MH17” - detail map - south-east Asian



(c) “PokémonGo!” - world map

Figure 11.3: A spatio-temporal trend dissemination example.

nation pattern. Curiously, once again, south-east Asia is a strong hub for this trend, resulting from a local burst on this topic from Indonesia.

But rather than looking at a few, hand-selected, trends as shown in these figures, we use our trend mining algorithm introduced in Part II to automatically extract the disseminations of a large number of past trends. Each trend yields a spatio-temporal trend-tensor, containing for each discrete time interval and each spatial region the number of corresponding tweets. As our first contribution, we postulate and verify the hypothesis that trends follow different archetypes, which differ strongly in terms of their dissemination patterns. Using a clustering approach, we identify these archetypes trends. For new trends, this result can be used to quickly classify a new trend as an archetype trend, to more effectively predict its future dissemination, thus allowing to predict where a trend will move to in the near future.

To model the dissemination of trends in space and time, this chapter is organized as follows. The next section, Section 11.4 gives an overview over the state-of-the-art of modelling trends in space and time. Section 11.2, formally defines a trend, and introduces our notion and data structures to define the spatio-temporal motion of a trend. Section 11.3.3 presents our technical concept for modeling the dissemination of a trend. This concept is experimentally evaluated in Section 13.5 and concluded in Section 11.6.

11.2 Preliminaries

This section will define terms and notations used throughout this work, and formally defines the problems tackled in the following. In this chapter we consider spatio-temporal text data, that is text data annotated with a geo-location and a timestamp, such as obtained from Twitter.

Definition 1 (Spatio-Temporal Text Database)**.** *A spatio-temporal text database $\mathcal{DB}$ is a collection of triples (s, t, c), where s is a point in space, t is a point in time, and c is a textual content.*

A concept that we adopt from the literature is the concept of a *trend* as introduced in [164].

Definition 2 (Trend)**.** *A trend $\tau_{K,t}$ is a set of keywords K that appear significantly more often starting at a time t.*

A more formal definition, which introduces the requirements of a set of terms to be considered as significant, please review Section 5.4 in Part II of this thesis. The set of spatio-temporal text objects which support trend $\tau_{K,T}$, is denoted as

$$\mathcal{DB}_{\tau_{K,T}} = \{(s, t, c) \in \mathcal{DB} | c \in K \wedge t \in T\}.$$

Definition 3 (Spatio-Temporal Occurrence)**.** *Let $\tau_{K,T}$ be a trend. Let $\mathcal{S} = \{S_1, ..., S_{|S|}\}$ be a partitioning of space into spatial regions, and let $\mathcal{T}$ be a partitioning of time into equi-sized time intervals denoted as epochs. Further, let $T := t \cap \mathcal{T} = \{T_1, ..., T_{|T|}\}$ be the set of epochs overlapping the trending time T. Then*

$$Occ_{\tau_{K,T},S} = |\{(s, t, c) \in \mathcal{DB} | s \in S \wedge t \in T \wedge c \in K\}|.$$

is the number of occurrences of trend $\tau_{K,T}$ at region S.

The aim of this work is to find the dissemination of trends, that is, pairs of spatial locations (S_1, S_2) such that any trend that appears in region S_1 is significantly more likely to appear in S_2 in the next epoch.

To describe the motion of a trend (K, t) in space and time, each trend is described by a time-space matrix, describing for each spatial region and each epoch $t \in T$ the number of tweets of the trend.

Definition 4 (Trend Count Matrix)**.** *The trend count matrix $D(\tau_{K,T}) \subseteq R^{|S|} \times R^{|T|}$ contains all occurrences of trend $\tau_{K,T}$ over space and time, and is defined as follows:*

$$D(\tau_{K,T})_{i,j} = Occ(\tau_{K,T_i}, S_j)$$

In this work, the main task is to analyze and mine multiple trend count matrices as defined in Definition 4, in order to identify groups of similar trends, groups of similar spatial regions, and to find common spatio-temporal dissemination of trends. These problems are formally defined as follows.

Definition 5 (Trend Clusters)**.** *Let $\mathcal{DB}$ be a spatio-temporal text database, let $\mathcal{DB}_\tau$ be a set of trends mined from $\mathcal{DB}$, and let $D(\tau \in \mathcal{DB}\tau)$ denote the trend count matrix of each trend. A trend cluster $C \subseteq \mathcal{DB}_\tau$ is a set of trends that exhibit mutually similar trend count matrices.*

Given a set of trends, the main challenge is to find association rules of the form "any trend observed in region A today, is likely to appear in region B tomorrow". This kind of spatio-temporal trend dissemination is defined as follows.

Definition 6 (Spatio-Temporal Trend Dissemination Rule)**.** *Let $\mathcal{DB}_\tau$ be a set of trends and their corresponding trend count matrices $D(\tau \in \mathcal{DB}\tau)$. For two spatial regions S_s and S_t, a spatio-temporal trend dissemination rule $S_s \to S_t$ implies that a large trend count at source region S_s at any time t indicates a large trend count at target region S_t at time $t+1$, formally:*

$$(S_s \to S_t) \leftrightarrow \forall i, \forall \tau \in \mathcal{DB}_\tau : D(\tau)_{i,s} \to D(\tau)_{i+1,t},$$

where $D(\tau)_{i,s} \to D(\tau)_{i+1,t}$ denotes that a large value in $D(\tau)_{i,s}$ implies a large value in $D(\tau)_{i+1,t}$

Finally, Definition 6 allows us to define the problem of spatio-temporal trend dissemination rule mining.

Definition 7. *Let $\mathcal{DB}_\tau$ be a set of trends and their corresponding trend count matrices $D(\tau \in \mathcal{DB}\tau)$. The problem of spatio-temporal trend dissemination rule mining is to find all pairs of spatial regions (S_s, S_t) such that $(S_s \to S_t)$ holds.*

11.3 Spatio-Temporal Trend Dissemination Rule Mining

This section describes our approach at mining spatio-temporal trend dissemination rules. As a first step, we need to acquire past trends, to mine dissemination rules from. For this purpose, we apply our event detection algorithm introduced in Part II. This allows us to reduce the otherwise infeasible amount of textual data to a relatively small set of remaining terms that correspond to significant trends and topics. On this remaining candidates and their corresponding tweets, we employ a space composition scheme in Section 11.3.1 to ensure having a similar number of tweets in each spatial region using a k-d tree. As a third step, we model the flow of trends over space and time in Section 11.3.2. Therefore, we transform a *trend count matrix*, as defined in Definition 4, into a *trend flow tensor*, which describes the flow from any source region to any target region at any point in time for any trend. Consequently, constructing a *trend flow tensor* for each trend that we have detected in the first step, yields a four-mode space $\times$ space $\times$ time $\times$ trends tensor, which will be fed to our fourth step, the mining step. In the mining step proposed in Section 11.3.3, we employ a tensor factorization approach to discover latent features of trends, latent features of trend-source-regions and latent features of trend-target-regions. These latent features allow us to cluster trends into sets of trends which disseminate similarly over space and time. Then, for each cluster of similar trends, we obtain trend flows from the *reconstructed trend flow tensor*.

11.3.1 Space Decomposition Scheme

To fit a flow model between spatial regions, we need to minimize the bias that results from having a non-uniform distribution of tweets on earth. We solve this problem by partitioning the geo-space in a way that minimizes the difference of tweets between spatial regions. For this purpose, we insert the geo-locations of all tweets in our database into a k-d tree, having a maximum node capacity of $1,000$. Thus, every leaf node of this k-d tree is guaranteed to have between 500 and $1,000$ two-dimensional points. Each of this leaf nodes is then used as a spatial region in the remainder of the work. The decomposition that we obtained this way is exemplarily shown in Figure 11.4. Note that this tree is constructed upon a typical, yet static, set of tweets.

11.3.2 Trend Flow Modeling

In this section we describe our approach of obtaining a trend flow from raw trend s. Thus, for a given trend, we consider all N occurrences of this trend at some time t and all M occurrences at the next time $t + 1$. All the regions having the trend at time t can be considered as sources of the trend, and all regions having the trend at time $t+1$ can be considered as targets of the trend.

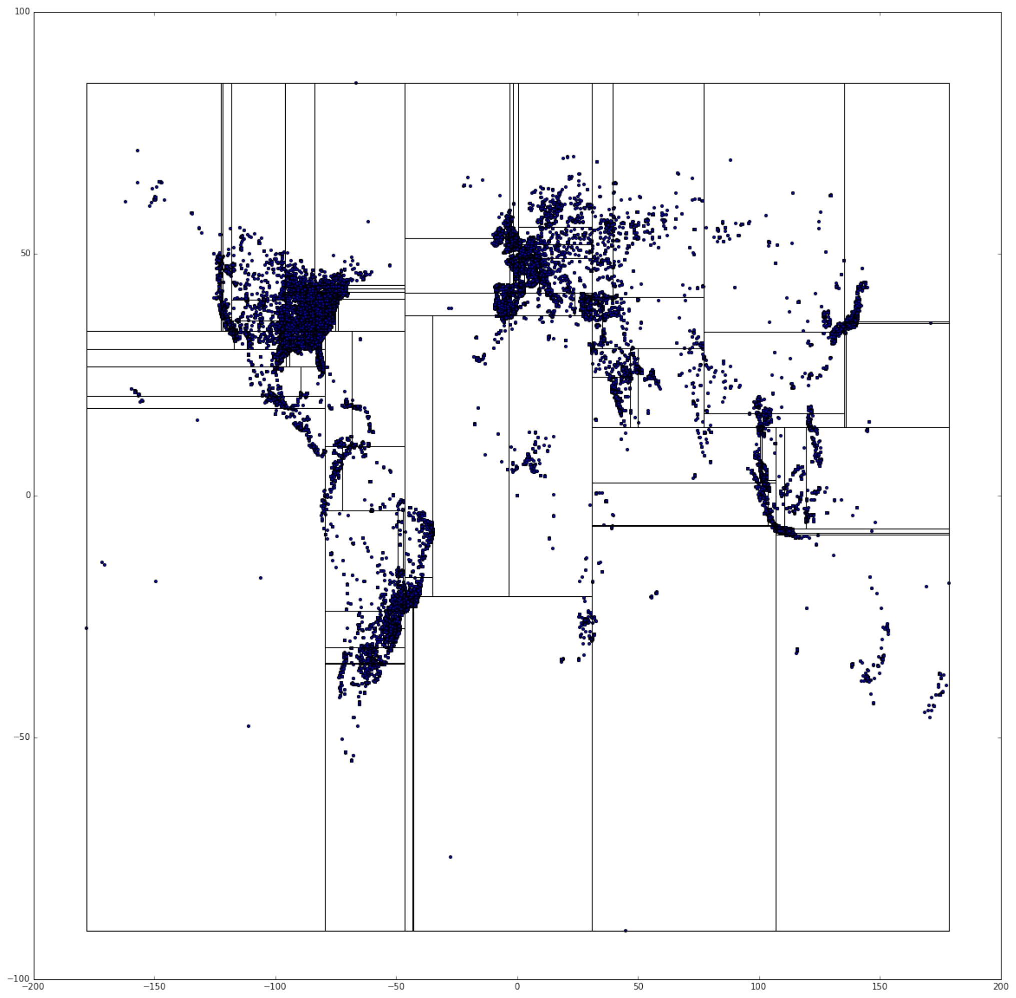


Figure 11.4: A k-d tree based space decomposition

Yet, we do not know any more specifically, which source region has affected which target region and to what degree, since we do not know through which channels and medias the trend was disseminated. Thus, due to lack of better knowledge, we assume that all sources affect all target uniformly. This flow model is formalized as follows

Definition 8 (Spatio-Temporal Trend Flow Model)**.** *Let* $\tau_{K,T}$ *be a trend having a set of keywords* K *and having a time interval* $T = \{T_1, ..., T_{|T|}\}$ *which covers* $|T|$ *epochs. Let* $\mathcal{S} = \{S_1, ..., S_{|\mathcal{S}|}\}$ *be a space composition into* $|\mathcal{S}|$ *spatial regions. Furthermore, let* $D(\tau_{K,T})_{i,j} = Occ(\tau_{K,T_i}, S_j)$ *be the trend matrix of* $\tau_{K,T}$*. We define the* trend flow model $F(\tau_{K,T})$ *of trend* $\tau_{K,T}$ *as a* $\mathcal{S} \times \mathcal{S} \times \{T_1, ..., T_{|T-1|}\}$ *tensor, such that*

$$F(\tau_{K,T})_{i,j,k} = \frac{Occ(\tau_{K,T_k}, S_i) \cdot Occ(\tau_{K,T_{k+1}}, S_i)}{\sum_{S_n \in \mathcal{S}} Occ(\tau_{K,T_{k+1}}, S_n)}$$

Intuitively, an entry $F(\tau_{K,T})_{i,j,k}$ of tensor $F(\tau_{K,T})$ corresponds to the absolute flow of occurrences from region S_i to region S_j from time T_k to time T_{k+1}.

$T_i \rightarrow T_{i+1}$

$$\begin{pmatrix}2\\0\\0\\1\end{pmatrix} \xrightarrow[S_s \rightarrow S_t]{flow} \begin{pmatrix}3\\1\\0\\5\end{pmatrix} \xRightarrow[F(\tau_{K,T})]{tensor} \begin{pmatrix}\frac{6}{9} & \frac{2}{9} & 0 & \frac{10}{9}\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ \frac{3}{9} & \frac{1}{9} & 0 & \frac{5}{9}\end{pmatrix}$$

$T_{i+1} \rightarrow T_{i+2}$

$$\begin{pmatrix}3\\1\\0\\5\end{pmatrix} \xrightarrow[S_s \rightarrow S_t]{flow} \begin{pmatrix}2\\1\\3\\4\end{pmatrix} \xRightarrow[F(\tau_{K,T})]{tensor} \begin{pmatrix}\frac{6}{10} & \frac{3}{10} & \frac{9}{10} & \frac{12}{10}\\ \frac{2}{10} & \frac{1}{10} & \frac{3}{10} & \frac{4}{10}\\ 0 & 0 & 0 & 0\\ 1 & \frac{5}{10} & \frac{15}{10} & 2\end{pmatrix}$$

Figure 11.5: Trend flow modelling

Example 1. *To illustrate the construction of tensor $F(\tau_{K,T})$, consider an example depicted in Figure 11.5. Here, the occurrences matrix of a tensor of a trend is shown for four spatial regions. At the first point of time t_i, the trend appears twice in the first region and once in the fourth region, yielding the vector $(2, 0, 0, 1)^T$. The second t_{i+1} and third point of time t_{i+2}, the distribution of occurrences is $(3, 1, 0, 5)^T$ and $(2, 1, 3, 4)^T$, respectively, yielding the trend matrix shown in Figure 11.5. Transitioning from the first epoch t_i to the second epoch t_{i+1}, the occurrences change from $(2, 0, 0, 1)^T$ to $(3, 1, 0, 5)^T$. The first spatial location S_1, having an initial value of two tweets, is thus a source of the trend. Since we cannot observe the latent means of dissemination of a trend (through the internet, via TV, radio, word-of-mouth, etc.), we estimate that region S_1 disseminates its trend to all other regions having this trend. Since a fraction $\frac{3}{9}$ of all tweets at time t_{i+1} are observed in region S_1, we estimate a trend-from of $\frac{2\cdot3}{9}$ from region S_1 to itself. In contrast, only one trending tweet is observed at location S_2 at time t_{i+1}, of which we contribute a flow of $\frac{2\cdot1}{9}$ to S_2. Similarly, a flow of $\frac{1\cdot5}{9}$ is contributed from S_4 to S_4.*

It is notable that each time-slice of tensor $F(\tau_{K,T})$ is a rank-1 matrix, as all lines are multiples of each other. This redundancy is desirable, as it evenly distributes the flow from all source regions to all target regions, and this redundancy will be removed in a later tensor factorization step. For each trend $\tau_{K,T}$ we obtain a three-mode tensor as described in Definition 8. Concatenating these tensors for each trend $\tau \in \mathcal{DB}_\tau$ yields a four-mode tensor $\mathcal{F}(\mathcal{DB})$ which is passed into the trend flow mining step described in the following.

11.3.3 Trend Flow Mining

We propose to decompose tensor $\mathcal{F}(\mathcal{DB}) \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ using a CANDECOMP/ PARAFAC (CP) tensor decomposition [45], [79] using k latent features, where

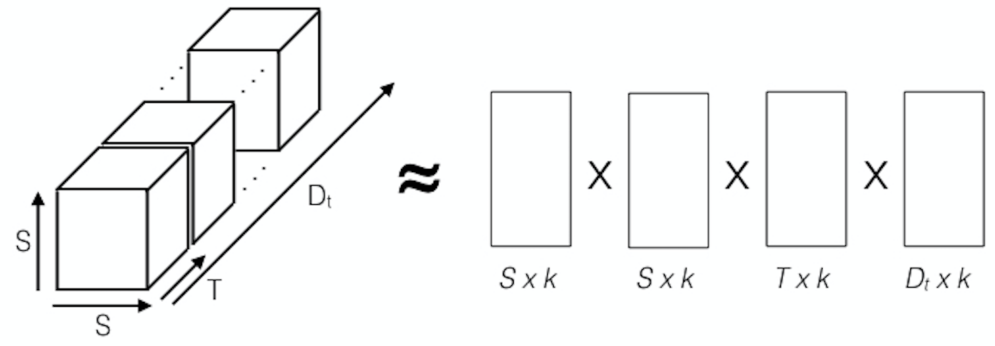


Figure 11.6: Trend Flow Modelling - Tensor Decomposition

k is a parameter of our algorithm. A CP factorization decomposes a tensor into a sum of component rank-one-tensors, i.e.

$$\mathcal{F}(\mathcal{DB}) \approx \sum_{r=1}^{k} u_r^1 \circ \cdots \circ u_r^N$$

where $u^n \in \mathbb{R}^{I_n}$ for $n = 1, \ldots, N$. Hence, as illustrated in Figure 11.6, this factorization decomposes our four-mode $\mathcal{S} \times \mathcal{S} \times T \times \mathcal{DB}_\tau$ tensor into four sets of vectors:

- a set of k vectors of latent features of length $|\mathcal{S}|$ describing each source spatial region,
- a set of k vectors of latent features of length $|\mathcal{S}|$ describing each target spatial region,
- a set of k vectors of latent features of length $|T|$ describing each time epoch, and
- a set of k vectors of latent features of length $|\mathcal{DB}_\tau|$ describing each trend.

These k-dimensional feature vectors can be used to identify mutually similar source spatial regions, mutually similar target spatial regions, mutually similar points in time, and mutually similar trends.

11.3.4 Trend Archetype Clustering

In our first mining step, we identify clusters of mutually similar trends, i.e. trends which have a similar feature vector after the factorization, and thus, since the tensor $\mathcal{F}(\mathcal{DB})$ describes the flow of trends over time, exhibit a similar dissemination over space and time. Each of the resulting clusters is called a trend archetype. This approach allows to classify future trends among all archetypes, and allows to predict the future dissemination of a new trend by using the dissemination model of their archetype.

Definition 9. *Let $\mathcal{DB}_\tau$ be a set of trends, and for each trend $\tau \in \mathcal{DB}_\tau$ let feat(τ) be a set of features describing τ. Further, let $\mathcal{C}(\mathcal{DB}_\tau) = \{C_1, \ldots, C_n\}$ be a clustering of all trends in $\mathcal{DB}_\tau$ into n clusters. Then we denote each cluster $C \in \mathcal{C}$ as an archetype, and all trends $\tau \in C$ are said to belong to the same archetype.*

11.3.5 Trend Archetype Flow Modelling

After the trend clustering step of Section 11.3.4, we can identify sets of trends which belong to the same dissemination archetype. Therefore, we return to the full tensor $\mathcal{F}(\mathcal{DB})$, and for each archetype $C \in \mathcal{C}$, we select only the trends $\tau \in C$, thus yielding a $\mathcal{S} \times \mathcal{S} \times T \times C$ tensor $\mathcal{F}(\mathcal{DB}, C)$ for each archetype C. Using $\mathcal{F}(\mathcal{DB}, C)$, we perform a projection on two modes $\mathcal{S} \times \mathcal{S}$ by averaging over all trends $\tau \in C$ and all epochs $T_i \in T$ to obtain the flow model of archetype C.

11.4 Related Work and Discussion

In contrast to the traditional related work for Event Detection, as discussed in Part II, we will exploit the spatio-temporal characteristics of an event. Unankard et al. [185] extracted user locations and event locations from geo-tagged posts. They defined a location correlation score between user and event locations and used it to identify the hotspot events. Zhou et al. [206] extended the Latent Dirichlet Allocation (LDA) to incorporate the location information of social messages, and proposed a novel location-time constrained topic model. Then they detected events by conducting similarity joins in streams of social messages. Sakaki et al. [158] conducted semantic analysis in user posts to detect natural disasters. They used exponential distribution to study the temporal characteristics of disasters. They used a Kalman filter and particle filter to predict the spatial trajectories of disasters. From the perspective of query processing, Lappas et al. [110] defined two types of spatio-temporal burstiness patterns, aiming at finding terms which had unusually high frequencies in a spatial region within a particular time interval. Sankaranarayanan et al. [160] developed a news system based on Twitter streams. They used Naive Bayes classifier to distinguish valuable news from junk posts and used an algorithm called leader-follower clustering to cluster news into topics. Appice et al. [26] proposed a technique where trend clusters are used to summarize sensor readings. However, such clusters consist of sensor entities themselves as opposed to trends.

Table 11.1: Trend Archetypes of 2014

#	Size	Example 1 Keywords	Example 2 Keywords
1	8	mh17 malaysia_crash	ferguson michael_brown riot
2	3	ellen degeneres selfie	robin williams suicide
3	5	whatsapp facebook takeover	supreme_court obergefell hodges
4	10	germany fifa14 brazil	germany fifa14 argentina
5	4	brazil world_cup	ebola
6	12	eu_sanction eu_russia	putin peskov conference
7	1	chile iquique earthquake	-
8	10	flappy_bird removed_appstore	how_I_met_your_mother_finale
9	18	mh370 malaysia_missing	qz8501 air_asia missing
10	14	scotland independence_poll	india bharatiya janata election
11	14	sydney siege hostage	ottawa gunman parliament
12	1	merry chistmas	-

11.5 Experimental Evaluation

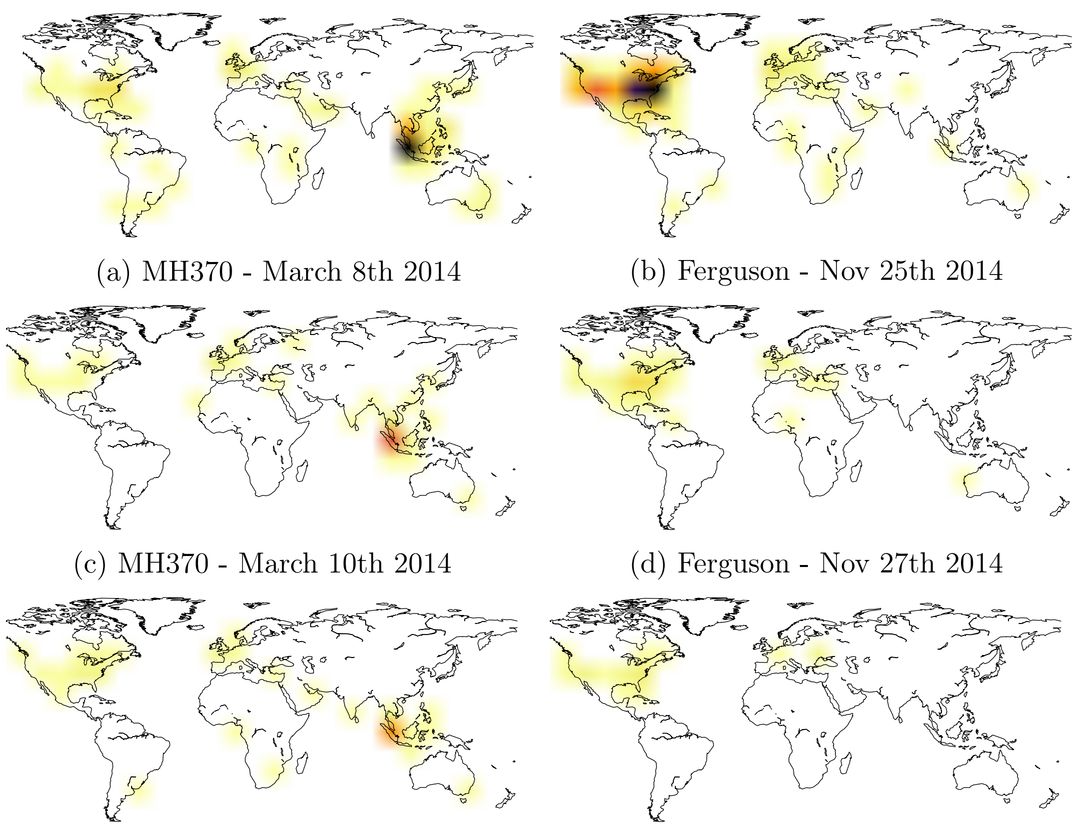
11.5.1 Parameters and Data Set

We evaluated our proposed workflow on the same Twitter data set that we introduced in the previous parts of this thesis. Tweets were aggregated over one-day periods by their UTC timestamp. The number of tweets per day ranged from around 50,000 to 150,000.

For each trend candidate extracted from our event detection algorithm, we extracted the corresponding original tweets from one day before and five days after the respective associated date to cover the entire trend dissemination pattern. Unless otherwise specified, each day was subdivided into epochs of six hours to allow for timeshift in different hemispheres. For the majority of our experiments, we used the top-100 trends of the year 2014.

11.5.2 Evaluation of Trend Archetypes

Table 11.1 depicts some exemplary resulting trend archetypes from data covering the year 2014. Keywords for the top-100 trends were extracted with our Event Detection algorithm from Part II and used to filter geo-tagged tweets occurring within a 5 day timeframe around the trend date. Underscores "_" between words denote a boolean conjunction, requiring all connected words



(a) MH370 - March 8th 2014 (b) Ferguson - Nov 25th 2014

(c) MH370 - March 10th 2014 (d) Ferguson - Nov 27th 2014

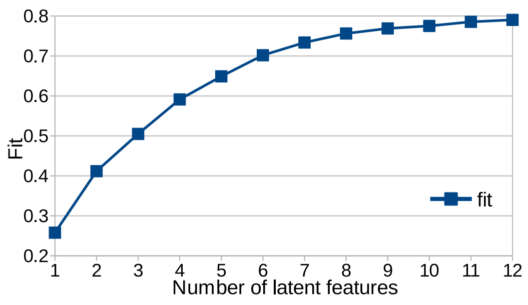
(e) MH370 - March 12th 2014 (f) Ferguson - Nov 29th 2014

Figure 11.7: Dissemination of trends “MH370” and “Ferguson”

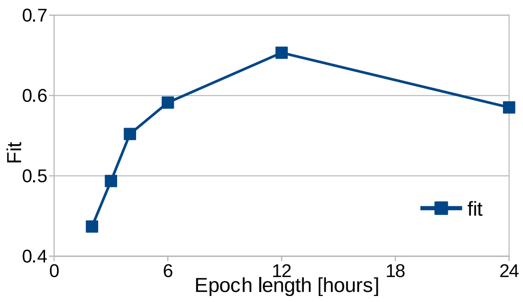
to occur in any possible order within one tweet. Spaces between keywords or conjunctions of keywords denote a boolean disjunction. Keywords listed are not exhaustive.

Each line of the table corresponds to a resulting archetype of trends with similar dissemination, resulting from a clustering of the latent feature vector feat(τ). While column “Size” refers to the true cardinality of each cluster, (up to) two examples are given to illustrate the nature of each archetype. Each example lists some keywords for one trend grouped into this archetype.

Some rather interesting results emerge by comparing the keywords to their respective historical events. While archetype #9 contains two trends referring to airplanes going missing without a trace (MH370 in March and QZ8501 in December), another lost airplane is grouped together with riots in the aftermath of a police shooting in the US in archetype #1. Looking at the respective tweet heatmaps in Figure 11.7, a similarity in pattern emerges: a first main event occurs (“plane crashes in Ukraine” vs. “riots after jury decision not to indict shooter”) causing an initial burst mainly in the affected areas (Figures 11.7a for MH370 and 11.7b for the shooting). After the initial burst, new information sheds different light on the events, making them stand out and causing a more steady flow of messages internationally (“plane grounded by



(a) Fit for number of latent features.



(b) Fit for length of trend episodes.

Figure 11.8: Approximation fit of factorized tensor.

missile" vs. "several people killed as riots spread"). This more steady output can be seen over Figures 11.7c and 11.7e for MH370 and Figures 11.7d and 11.7f for the shooting. Bear in mind that the grouping occurred solely based on the numerical features of the respective trends' spatial dissemination, regardless of their content.

Trend archetype #2 grouped some strong international trends related to society, containing Ellen DeGeneres' selfie picture taken at the Oscar ceremony as well as Robin Williams' sudden suicide. Archetype #3 contains trends with more specialised contents such as financial ("Facebook buys WhatsApp") or judicial ("Obergefell vs. Hodges, Supreme Court deciding on same-sex marriage").

Another distinction is made between archetypes #4 and #5, both containing trends regarding the FIFA world cup 2014 in Brazil: while #4 represents game results and surprising or strong wins, #5 contains the more steady general discussion about the event, as well as other longer–term themes sparking much discussion. Among those is also the repeated outbreak of the Ebola virus in West Africa. Despite the entirely different nature of those topics, both represent a great public interest that dominated news media for longer periods of time.

11.5.3 Evaluation of approximation quality

The tensor decomposition employed in our flow modelling process exhibits a high quality for even low numbers of k, i.e., a small number of latent features per feature vector. This indicates large eigenvalues of the first k latent features, thus indicating that these features are able to accurately describe the whole tensor with little loss of information. However, some information is still lost compared to an undecomposed tensor. We evaluate the quality of our decomposition by summing up the least-squared error between a reconstruction of the original tensor from its k-feature-vectors, and the original tensor itself. We call the inverse of this error "fit", ranging from 1.0 for an exact match to 0.0 for no correlation.

Figure 11.8a shows that for a $k = 4$: the reconstructed tensor matches its

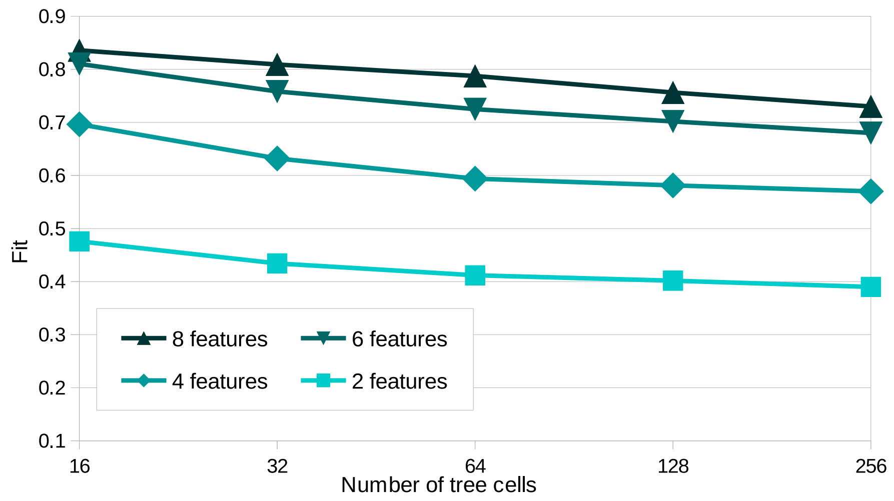


Figure 11.9: Fit over tree cells for varying latent features.

original with a fit of 0.6, which is why we chose to set $k = 4$ in all subsequent experiments unless otherwise specified. As we can see, the gain in fit slows down with additional latent features.

Figure 11.8b displays fit for different lengths of trend epochs, the granularity of our analysis in temporal dimension, ranging from 2 hours to 24 hours. The amount of days looked at per trend remained the same, so a longer epoch will result in a smaller number of epochs overall, reducing the size of $\mathcal{F}(\mathcal{DB})$ in the T dimension. Intuitively, a smaller tensor $\mathcal{F}(\mathcal{DB})$ is easier to reconstruct, increasing the fit for longer epochs. However, this does not hold for epochs of 24 hours. We believe this to be due to a counter effect of more diversity in tree cell population as epochs get longer and thus more tweets are grouped in the same epoch. In other experiments, we set the epoch length to 6 hours unless otherwise specified – although it is not the peak for fit, we found it to best approximate trends from different global regions, hence being able to compare trends in different hemispheres where peaks happen at different hours in the day.

The effect of varying spatial resolution can be seen in Figure 11.9 for four alternative settings of k. Although the underlying k-d tree is built on global tweet distribution to assure tweets in the same region from different trends are matched to the same cell, varying its node capacity upon indexing results in a higher- or lower-resolved spatial grid, hence lowering or increasing the size of $\mathcal{F}(\mathcal{DB})$ in both spatial dimensions. Naturally, a smaller grid is easier to approximate with the same amount of latent features, yet the experiments show that features have a much higher impact on approximation quality than changing spatial resolution. As we can see, fit values do not deteriorate much for higher numbers of grid cells.

The impact of different numbers of trends $\tau_{K,T}$ is stronger, particularly for smaller k. Figure 11.10 displays fit values for four alternative settings of k and

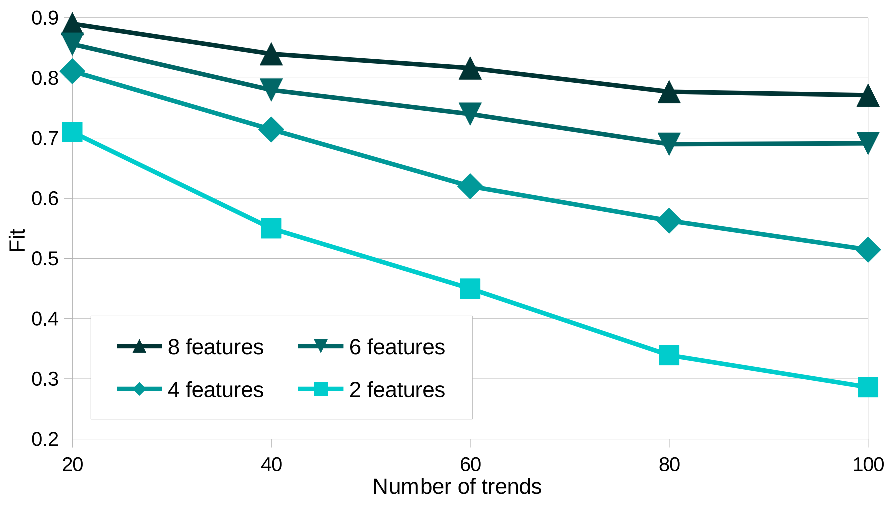


Figure 11.10: Fit over trends for varying latent features.

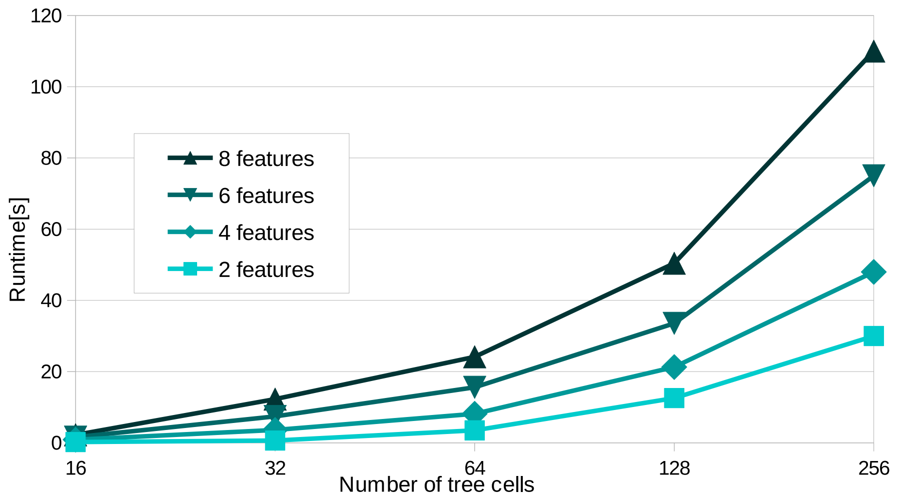


Figure 11.11: Runtime over tree cells for varying latent features.

the number of trends ranging from 20 to 100. As in previous experiments, fit decreases as the size of $\mathcal{F}(\mathcal{DB})$ increases. However, for higher k the effect is drastically smaller, maintaining a good approximation quality at the cost of a higher complexity.

11.5.4 Evaluation of algorithmic runtime.

The following experiments evaluate runtime of the tensor generation, decomposition and projection on two modes $S \times S$. Filtering of tweets is not included in this evaluation since it depends heavily on the actual keyword settings as well as size of the underlying data set. All experiments were performed on Arch Linux on an Intel i7 notebook with 16 GB of memory, implemented in the Python language using `numpy`, `pandas`, and the `sktensor` package for tensor

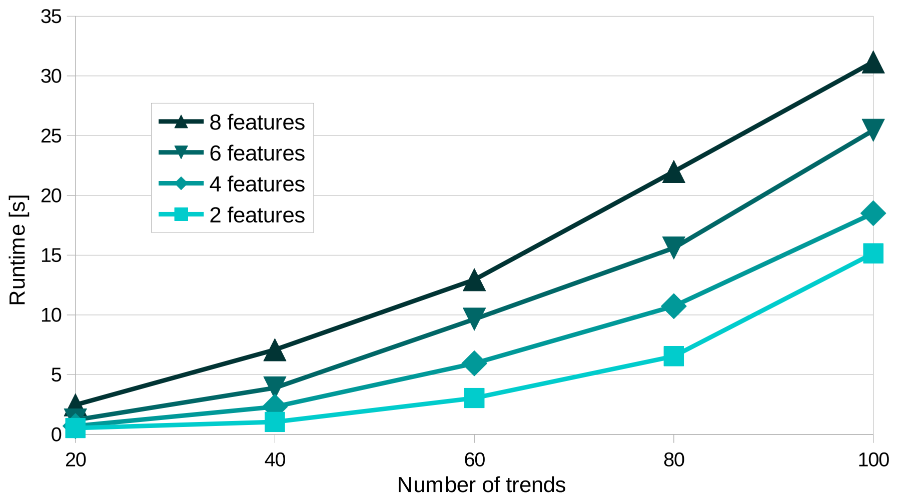


Figure 11.12: Runtime over trends for varying latent features.

decomposition.

Figure 11.11 examines runtime in seconds over spatial resolution, for four different settings of k. Since an increase in the number of tree cells causes a quadratic increase in the size of $\mathcal{F}(\mathcal{DB})$, runtime scales superlinear for higher spatial resolutions.

The effect of different numbers of trends $\tau_{K,T}$ on runtime is shown in Figure 11.12. Runtimes show only a slight superlinear increase for higher amounts of trends, as the size of $\mathcal{F}(\mathcal{DB})$ increases linearly with trends.

11.6 Conclusions

In this chapter, we studied the dissemination of trends in space and time. For each trend obtained from our Event Detection algorithm of Part II, we proposed to constructed a spatio-temporal trend dissemination model to describe the flow of a trend through space and time. By applying a tensor factorization approach, we extracted latent features of trends, to which we applied a clustering approach to obtain sets of trends having a similar dissemination archetype. Our qualitative evaluation of these trend archetypes on Twitter trends show meaningful dissemination archetypes, such as political trends, celebrity trends, and disaster trends. Our quantitative analysis shows that our tensor factorization yields are high approximation quality for a low number of latent features. This result implies that a small number of latent features we derive from the flow of each trend is able to discriminate trends with a high-precision.

The algorithm outlined in this chapter can be included in a system to classify trend archetypes immediately after they have been discovered by our Event Detection algorithm.

Chapter 12

Event Co-location Mining

In this chapter we will show how to make use of the underlying social network of users within our Event Detection pipeline. We introduce geo-social co-location mining, the problem of finding social groups that are frequently found at the same location. Furthermore, this problem has applications in social sciences, allowing to research interactions between social groups and permitting social-link prediction. It can be divided into two sub-problems. The first sub-problem of finding spatial co-location instances, requires to properly address the inherent uncertainty in geo-social network data, which is a consequence of generally very sparse check-in data, and thus very sparse trajectory information. For this purpose, we propose a probabilistic model to estimate the probability of a user to be located at a given location at a given time, creating the notion of probabilistic co-locations. The second sub-problem of mining the resulting probabilistic co-location instances requires efficient methods for large databases having a high degree of uncertainty. Our approach solves this problem by extending solutions for probabilistic frequent itemset mining. Our experimental evaluation performed on real (but anonymized) geo-social network data shows the high efficiency of our approach, and its ability to find new social interactions.

12.1 Introduction

Spatial features describe the presence or absence of geographic object types at different locations. Examples of spatial features include plant species, animal species, road types, cancers, crime, and business types, or features of individuals, such as personal preferences, or simply their ID. A spatial co-location pattern represents a subset of spatial features whose instances are frequently located in a spatial neighborhood. For example, "botanists may have found that there are orchids in 80% of the area where the middle-wetness green-broad-leaf forest grows" (example taken from [190]). Spatial co-location patterns may yield important insights for many applications. For example, a mobile service provider may be interested in services frequently requested by geographical neighbors, and thus gain sales promotion data. Other application domains include Earth science, public health, biology, transportation and geo-social networks. Traditional solutions for the problem of frequent co-location mining [190] consider classical spatial data, where each data record has a spatial location.

In this chapter we take the problem of spatial co-location mining into a new context, by considering spatio-temporal data, i.e., trajectory data of individuals. Thus, the problem now is to find groups of users which frequently co-locate in geo-space over time, creating the notion of geo-social co-location mining. There is already an abundance of public data sets that can be mined, including data sets from geo-social networks [54] and from social networks using geo-tags such as Twitter. Frequent co-location mining on such data may yield interesting patterns, such as "Members of LMU and HKU are frequently to be found at the same location, while members of some other university are often found in solitude or among themselves". In such an application, each instance of a co-location corresponds to a (l, t, S) triple, where S denotes the set of individuals that have been at the same location l at the same time t. The problem of geo-social co-location mining introduces two major new challenges which have not been sufficiently covered in existing work on traditional co-location mining. Firstly, the temporal dimension leads to very large sets of co-location instances, since every location and time pair leads to a possibly non-empty co-location instance, secondly existing solutions do not consider the uncertainty which is inherent in spatial data: Spatial data may be imprecise (e.g., due to measurement errors), data can be obsolete (e.g., when the most recent position update is already minutes old), data may originate from unreliable sources (such as crowd-sourcing), or it may be blurred to prevent privacy threats and to protect user anonymity [55]. For example, the oval regions in Figure 12.1 may correspond to individual persons, while the color of each person may represent the individual's affiliations. Here, the location of each person is a conservative approximation based on the users GPS history. It is important to note that we are considering historic data. Thus, for a given point of time t, both past and future GPS positions of a user may be available.[1] Given these approximations, it becomes possible to estimate which point of interest each user is currently visiting, yielding probability distribution as shown in the table in Figure 12.1 for depicted point of time (22:00) and for a point of time one hour later. Given such data, we can immediately envision a number of useful applications:

- Find groups of people often co-locating. In the setting described above, two individuals being located at the same location may not actually be there together, but if the two individuals co-locate very often, it becomes highly unlikely that their co-locations are independent random events.

- Find groups of people visiting the same types of points of interest, even not at the same time. This allows to cluster user's by their points of

[1]A probabilistic model to estimate the position of a mobile user given past and future observations can be found in [144].

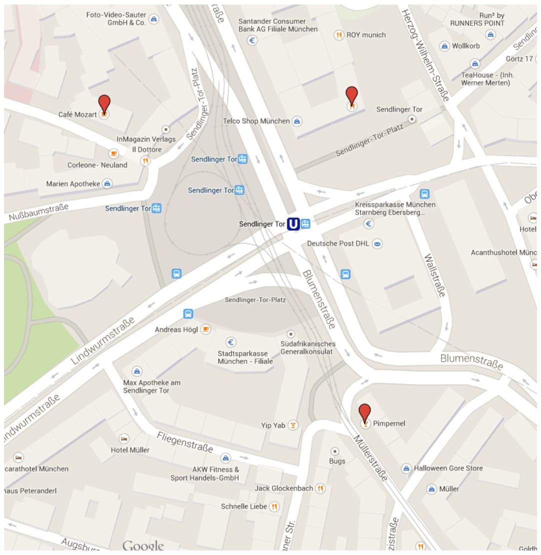


Figure 12.1: Spatial co-location mining in uncertain spatio-temporal data (Source: `http://maps.google.de`).

interest, thus allowing to predict new locations that a user might find interesting, based on other users in the same cluster.

- Classify points of interest by the people that visit these. For example, if a point of interest that is unlabeled is being visited by a significant fraction of users which (individually or even together) visit Italian restaurants, then it might be possible to predict the label of this point of interest.
- Visiting a new city, such as Melbourne, new people, that are similar to people you hang out with at home, and new point of interest, that are similar to locations you visit at home, can be predicted to you and to the people that you now hang out with in Melbourne.

12.2 Problem Definition

In traditional co-location mining, the location of an object is known for certain. Under this assumption, a lot of work has been published in the last decade [82, 83, 172, 81, 205]. A survey on the field of co-location mining on certain spatial data be found in [128, 129]. However, in many real applications such as plant disease diagnosis, environmental surveillance and geo-social networks, the location of objects is uncertain. In the following, the problem of probabilistic spatial collocation mining on uncertain spatial data is defined. To formally define the problem of spatial co-location mining, we first have to define the concept of a spatial co-location:

Definition 10 (Spatial Co-location Instance)**.** *Given a reflexive and symmetric neighbor relation $\mathcal{R}$ over a spatial database $\mathcal{DB}$, a* spatial co-location instance *is a set $I \subseteq \mathcal{DB}$ of spatial objects that form a clique [34] under the relation $\mathcal{R}$, i.e., $\forall o_1, o_2 \in I : (o_1, o_2) \in \mathcal{R}$.*

In this work, a neighbor relation with particular importance in social network applications will be used. This relation uses a set of interesting spatial locations, such as bars, restaurants and football stadiums. Two individuals are co-located if they are sufficiently close to the same location, formally.

Definition 11. *Let $\mathcal{L}$ be a set of spatial locations, and let $\mathcal{DB}$ be a database of spatial objects. The neighbor relation $\mathcal{R}$ is defined as follows*

$$(o_i \in \mathcal{DB}, o_j \in \mathcal{DB}) \in \mathcal{R} \Leftrightarrow \exists l \in \mathcal{L} : dist(o_i, l) \leq \epsilon \wedge dist(o_j, l) \leq \epsilon$$

An example of the problem of frequent co-location mining in uncertain spatial data using the neighborhood relation of Definition 11 is given in the following.

Example 2. *Consider uncertain positions of individuals in a geo-social network application. The task is to find groups of people that commonly spend time at the same locations, in order to predict missing links in the underlying social network, or to offer special deals to such groups. Figure 12.1 exemplarily shows the position of individuals $A, ..., G$, and three locations: a café, a restaurant and a bar. For simplicity, each of these locations is represented by an oval region, but in practice, these uncertainty regions can have arbitrary shapes [144]. It is not possible to tell for certain, whether user A is located inside the café, or just barely outside of it. In contrast, user D is certainly inside the restaurant, while user C is certainly outside all three places. The probability $P(U$ in $l)$ that a user U is located inside a location l can be computed using techniques for range queries on uncertain data [37].[2] Exemplary probabilities $P(U$ in $l)$ for all users U and all locations l are shown in the table of Figure 12.1. At time 22:00, the users E, F and G are co-located at the bar with a high probability. However, at time 23:00, user G is likely no longer located with users E and F.*

Clearly, the number of co-locations may be extremely large, since in an application like this, there may be one non-empty co-location for each combination of time stamp and location. Since the users' location readings are discrete in time, a global partitioning of the data set into separate time slots (e.g., 1 hour) makes sense. The task of probabilistic co-location mining is to find groups of users (objects), having a significantly high probability of having spent time at the same location for a sufficiently large number of times. Formally, the problem of probabilistic frequent spatial Co-location mining is defined as follows.

[2]For the case proximity to a location is not modelled by a circle, an adaption of the techniques in Section 12.4 can be made easily, by replacing distance calculation by intersection tests between points and polygons.

Definition 12. *Given a set $\mathcal{F} = \{f_1, ..., f_k\}$ of k spatial features, given a database $\mathcal{DB} = \{o_1, ..., o_N\}$ of N uncertain spatial objects each having a set $f(o_i \in \mathcal{DB}) \subseteq \mathcal{F}$ of spatial features, and given a positive integer minSup and a probability threshold τ, a probabilistic frequent spatial co-location mining algorithm returns all sets $S \subseteq \mathcal{F}$ of features such that the probability there exist at least minSup spatial co-locations instances I such that $S \subseteq \bigcup_{o \in I} f(o)$ is at least τ.*

To find probabilistic frequent spatial co-locations, consider the following example.

Example 3. *Returning to Example 2 assume that $minSup = 2$ and $\tau = 0.5$ and consider the spatial features $F = \{red, green, purple\}$, depicted by the corresponding colors in Figure 12.1, which may e.g., correspond to the affiliations of mobile users. In this example, we have two possible co-locations instances of features red and green, in the café and in the bar at times 22:00 and 23:00. Assuming independence between uncertain objects[3], the probability of a co-location of green and red at the café at time 22:00 can be computed by the product of marginal probabilities $P(A \wedge B) = P(A) \cdot P(B) = 0.4 \cdot 0.2 = 0.08$. At the bar at time 22:00, the probability of a co-location between red and green can be computed by $P(E \wedge (F \vee G)) = P(E \wedge \neg(\neg F \wedge \neg G)) = P(0.6 \cdot (1 - 0.3 \cdot 0.2)) = 0.564$. At time 23:00, can obtain the co-location probabilities red and green at the café and the bar of 0.9 and 0.588, respectively. Given these probabilities, we can compute the probability that at least $minSup = 2$ co-location instances exist by applying the generating functions technique of [118, 119], yielding a probability of 0.778 which is greater than $\tau = 0.5$. Thus the set of spatial features red and green will be returned as a probabilistic frequent co-location.*

In the following section, we propose solutions to compute the probabilities of probabilistic frequent co-locations efficiently.

12.3 Related Work

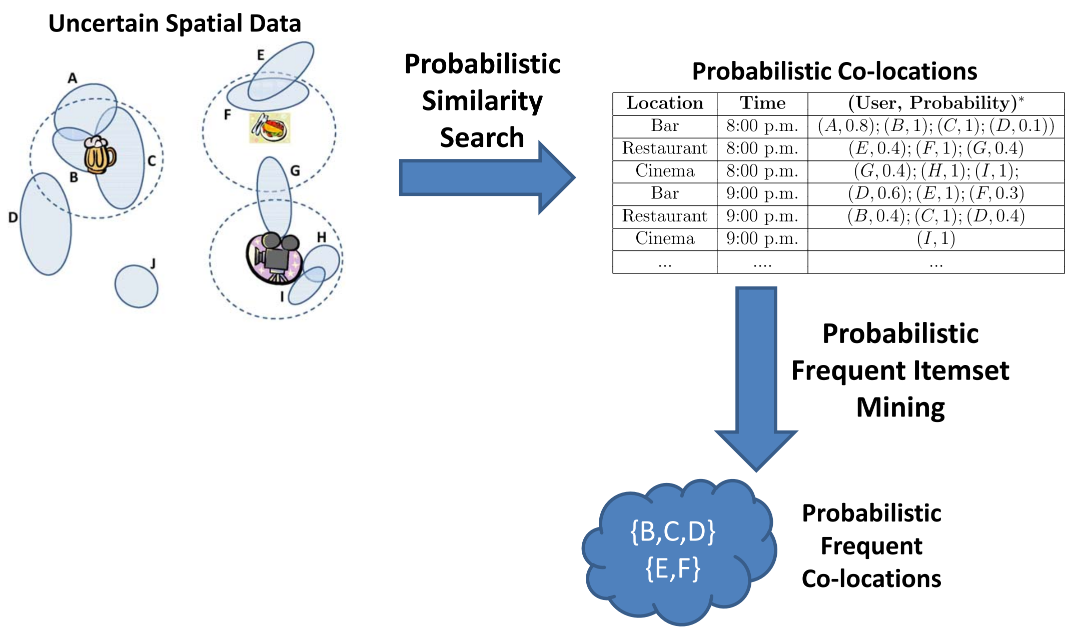
Traditional co-location mining on (certain) spatial data has been studied in the past [198, 205, 81]. These works define a spatial neighborhood relation on pairs of objects that do not exceed a given distance threshold. Due to the assumption of certain objects, the works can solve the problem of frequent co-location mining by applying traditional frequent pattern mining solutions such

[3]We argue that in many applications, this assumption holds true. Note that the position of mobile objects can be strongly correlated, as for example friends are more likely to travel together. However, the assumption that measurement errors are mutually independent does often hold. Thus, we assume GPS errors between different devices to be independent, and uncertainty regions that are added deliberately for privacy preservation should be independent as well. Nevertheless, this assumption of independent random variables of spatial locations can be a base for discussions.

as Apriori-algorithm [14] combine the discovery of spatial neighborhoods with the mining process.

The problem of probabilistic co-location mining in uncertain spatial data is related to the problem of frequent itemset mining in uncertain transaction databases. Existing solutions for this problem transform uncertain items into certain ones by thresholding the probabilities – for example, by treating all uncertain items with a probability value higher than 0.5 as being present, and all others as being absent in a transaction. Such an approach loses useful information and leads to inaccuracies. Existing approaches in the literature are based on expected support ([56, 57, 9]). Itemsets are considered frequent if the expected support exceeds *minSup*. Effectively, this approach returns an estimate of whether an object is frequent or not with no indication of how good this estimate is. Since uncertain transaction databases yield uncertainty w.r.t. the support of an itemset, the probability distribution of the support and, thus, information about the confidence of the support of an itemset is very important. This information, while present in the database, is lost using the expected support approach.

There is a large amount of research on Frequent Itemset Mining (FIM) but very little work addresses FIM in uncertain databases [56, 57, 117]. The approach proposed by Chui et. al [57] computes the expected support of itemsets by summing all itemset probabilities in their U-Apriori algorithm. Later, in [56], they additionally proposed a probabilistic filter in order to prune candidates early. In [117], the UF-growth algorithm is proposed. Like U-Apriori, UF-growth computes frequent itemsets by means of the expected support, but it uses the FP-tree [78] approach in order to avoid expensive candidate generation. In contrast to our probabilistic approach, itemsets are considered frequent if the expected support exceeds *minSup*. The main drawback of this estimator is that information about the uncertainty of the expected support is lost; [56, 57, 117] ignore the number of possible worlds in which an itemsets is frequent. [204] proposes exact and sampling-based algorithms to find likely frequent items in streaming probabilistic data. However, they do not consider itemsets with more than one item. To the best of our knowledge, our approach in [36] was the first that is able to find frequent itemsets in an uncertain transaction database in a probabilistic way. However, this publication has stimulated research on the field of probabilistic mining of frequent itemsets in uncertain transaction data, creating a large number of follow up publications. A detailed survey can be found in [183]. In [188, 189], an approach is presented to approximate the support PDF of an itemset using a Poisson distribution. This approach yields a very small error if the database is sufficiently large. This approximation furthermore allows to compute the support PDF of an item much faster than the exact approach presented in [36]. An approach to accelerate the computation of our approach in [36] was presented by [98], using massive parallelization exploiting GPGPU (General-Purpose computation on GPU). Furthermore, the related problem of mining frequent subgraphs over uncertain graphs [120, 214, 213] has gained a lot of research interest in


Uncertain Spatial Data
A
B
C
D
E
F
G
H
I
J
Probabilistic Similarity Search
Probabilistic Co-locations
Probabilistic Frequent Itemset Mining
{B,C,D}
{E,F}
Probabilistic Frequent Co-locations

Location	Time	(User, Probability)*
Bar	8:00 p.m.	$(A, 0.8); (B, 1); (C, 1); (D, 0.1))$
Restaurant	8:00 p.m.	$(E, 0.4); (F, 1); (G, 0.4)$
Cinema	8:00 p.m.	$(G, 0.4); (H, 1); (I, 1);$
Bar	9:00 p.m.	$(D, 0.6); (E, 1); (F, 0.3)$
Restaurant	9:00 p.m.	$(B, 0.4); (C, 1); (D, 0.4)$
Cinema	9:00 p.m.	$(I, 1)$
...	....	...

Figure 12.2: Workflow of probabilistic spatial co-location mining.

the last years. Finally, an approach for probabilistic frequent itemset mining on uncertain data avoiding multiple database scans incurred by the candidate generation step of [36] has been proposed in [35].

Only recently, the research community has tackled the challenge of spatial co-location mining in uncertain data. Recent work from [190] considers existential uncertainty in spatial data. In this model, each object has a probability to be present in the database. The solution of [190] has a run-time polynomial in the number of possible worlds, thus exponential in the number of uncertain objects. The reason for this high complexity is the neighborhood relation $R(.,.)$ used in [190] is arbitrary, i.e., this approach can be applied to any neighborhood relation. This fact makes efficient co-location mining hard: For three uncertain objects A, B and C, the predicates $R(A, B)$ and $R(B, C)$ are stochastically dependent, despite the assumption of independence between objects.

12.4 Probabilistic Frequent Co-Location Mining

In a nutshell, the problem of probabilistic co-location mining requires two subtasks to be solved, as illustrated in Figure 12.2:

- First, for each location l and each time interval t, probabilistic instances have to be computed and derived. We partitioned This requires to compute the probabilities of all objects, to be close to location l at time t. This task requires to utilize probabilistic similarity search methods on uncertain spatial data to derive the probability that a given object is a member of a co-location instance. For the neighbor relation given in Definition 11, this step requires to perform probabilistic range queries, using the locations $\mathcal{L}$ as query points. As a result of the first step, an uncertain spatial database is transformed into a probabilistic co-location database such as depicted in Figure 12.4.

- Second, all probabilistic co-location instances need to be mined in order to detect subsets of spatial features having a statistically significantly high probability to be co-located frequently in the database. For this subtask, we can assume that a database $\mathcal{DB}$ of probabilistic co-locations such as featured in Figure 12.4 is given as a result of solving the first subtask. Given such a database, the task of finding probabilistic frequent co-locations in such a database is equivalent to the problem of probabilistic frequent itemset mining [36] in uncertain transaction data. Both problems, of probabilistic mining of spatial co-locations in uncertain spatial data, as well the problem of probabilistic frequent itemset mining in uncertain transaction data, are formally defined in the following.

12.4.1 Occurrence Probability Estimation

At each time interval t we estimate the probability of a user u being at a certain location l based on geographical distance $d_{u,l}$, using either Euclidean or Haversine distance, the latter being more precise. The probability $P_{t,u,l}$ that a user u occurs at location l at time interval t is then given by

$$P_{u,l} = \frac{\rho\left(d_{u,l}\right)}{\sum\limits_{l_t \in L_{u,t}} \rho\left(d_{u,l_t}\right)},\ \rho\left(d_{u,l}\right) = \begin{cases} 0 & d_{u,l} \geq \tau_d \\ \frac{1}{\sigma\sqrt{2\pi}}\, e^{-\frac{(x-\mu)^2}{2\sigma^2}} & d_{u,l} < \tau_d \end{cases}$$

where ρ is the density (PDF) regarding a normal distribution $\mathcal{N}(\mu,\, \sigma^2)$ with $\mu = 0$ and $\sigma^2 = \frac{\tau_d}{3}$. Thereby τ_d denotes a distance threshold parameter (e.g. 100 meters) to cut the long tail of the probability distribution to 0 for each user with a distance $d > \tau_d$. We utilize this threshold in our implementation for a simple yet effective spatial index based on grid cells of the size τ_d. For a latitude ϕ and longitude λ of a user or location we determine the corresponding (x, y) grid cell as such $\left(\left\lfloor \frac{\phi}{tau_d} \right\rfloor, \left\lfloor \frac{\lambda}{tau_d} \right\rfloor\right)$. An example is shown in Figure 12.3: For the user U only locations within the neighbor cells ± 1 around the user's cell are candidates. Thereby locations that share the same cell with the user (e.g. location C) are certain hits because their distance must smaller than τ_d. Locations like A which are at least 1 complete cell away must not be considered as their distance to the user must be greater than τ_d.

12.4.2 Transformation to Probabilistic Frequent Itemset Mining

The definition of a uncertain co-location database can be mapped to the definition of an uncertain transaction database defined in [36].

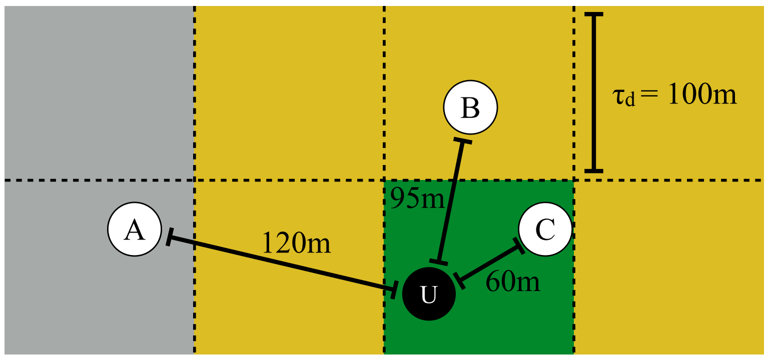


Figure 12.3: Grid based spatial index for efficient nearest neighbor queries

Definition 13 (Uncertain Transaction Database)**.** *Let I be a set of items. An* uncertain transaction *database $\mathcal{T}$ is a set of probabilistic transactions. Each transaction $T = \{i|i \in I, P(i)\} \in \mathcal{T}$ contains a set of items, each associated with a probability. For each pair $(i \in I, P(i))$, the probability $P(i)$ describes the likelihood that i is present in the probabilistic transaction T.*

This equivalence between Definition 12 and Definition 13 allows to interpret the problem of probabilistic frequent co-location mining in uncertain spatial data, as the problem of probabilistic frequent itemset mining in uncertain transaction data. This can be done by interpreting a spatial feature as an item or a probabilistic co-location instance as a transaction. Thus, solutions for the problem of probabilistic frequent item-set mining can now be applied. In fact, a large body of efficient algorithms (e.g. [188, 189, 98, 43]) have been proposed for the problem definition of [36]. Yet, a main common problem of these works is the lack of a real world application for the problem of probabilistic frequent itemset mining. We argue, that probabilistic frequent itemset mining and probabilistic spatial co-location mining can bridge this gap, thus providing spatial applications. In the following subsections, we will briefly outline a mapping of existing solutions to the problem of probabilistic co-location mining. Firstly, as a baseline a naive solution is presented, omitting the uncertainty information. Then, the exact solutions of [36] is reviewed and mapped to co-location mining. Finally, the same is done for the approximate solutions of [188]. For these solutions, an initial experimental run-time evaluation is presented in Section 13.5, by using a real-world data set consisting of geo-tagged tweets.

Naive Probabilistic Co-Location Mining

One *naive* approach is to transform an uncertain database into a non-uncertain database by setting the item probabilities to 0 or 1 and then applying a traditional frequent itemset detection method. For example, probabilities less then 0.5 could be mapped to 0 and probabilities above 0.5 could be mapped to 1.

ID	Co-location
t_1	(A, 0.8) ; (B, 0.2) ; (D, 0.5) ; (F, 1.0)
t_2	(B, 0.1) ; (C, 0.7) ; (D, 1.0) ; (E, 1.0) ; (G, 0.1)
t_3	(A, 0.5) ; (D, 0.2) ; (F, 0.5) ; (G, 1.0)
t_4	(D, 0.8) ; (E, 0.2) ; (G, 0.9)
t_5	(C, 1.0) ; (D, 0.5) ; (F, 0.8) ; (G, 1.0)
t_6	(A, 1.0) ; (B, 0.2) ; (C, 0.1)

Figure 12.4: Example of an uncertain co-location database where users are co-located at a certain location (e.g., restaurant)

However, such a transformation obviously involves loss of information and accuracy. Furthermore, we would have no idea how confident we could be in the results. In particular, itemsets that are often associated with probabilities close to 0.5 yield a very large error in the result. Another approach is to use the probabilities associated with the itemsets in order to compute the expected support of an itemset.

To avoid incurring a biased result, previous work was based on the expected support [56, 57, 117], i.e., the expected number of spatial co-locations of a group of spatial features.

Definition 14. *Given a set $\mathcal{F}$ of spatial features and a database of co-location instances I, the* expected support $E(X)$ *of a set of spatial features $X \subseteq F$ is defined as $E(X) = \sum_{i \in I} P(X \subseteq i)$.*

The expected support of set of spatial features X can be efficiently computed by a single scan over all co-location instances. An itemset is considered frequent if its expected support is above *minSup*. However, the later step has the major drawback that the uncertainty information is forfeited when using the expected support approach. Thus, information is lost about the likelihood that X is frequent.

Example 4. *As an example, consider the database depicted in Figure 12.4, containing a set of uncertain co-location instances. Treating each co-location instance as a transaction, the expected support of the itemset $\{D\}$ is $E(\{D\}) = 3.0$. The fact that $\{D\}$ occurs for certain in one transaction, namely in t_2, and that there is at least one possible world where X occurs in five transactions are totally ignored when using the expected support in order to evaluate the frequency of an itemset. Indeed, suppose $minSup = 3$; do we call $\{D\}$ frequent? And if so, how certain can we even be that $\{D\}$ is frequent? By comparison, consider itemset $\{G\}$. This also has an expected support of 3, but its presence* *or absence in transactions is more certain. It turns out that the probability that* $\{D\}$ *is frequent is* 0.7 *and the probability that* G *is frequent is* 0.91*. While both have the same expected support, we can be quite confident that* $\{G\}$ *is frequent, in contrast to* $\{D\}$*. An expected support based technique does not differentiate between the two.*

Concepts to evaluate the co-location instances in a probabilistic way are presented in the following.

Exact Probabilistic Support

A co-location is a *frequent co-location* if it occurs in at least *minSup* co-location instances, where *minSup* is a user specified parameter. The number of instances of a co-location is denoted as the support *supp*(S) of S. In uncertain co-location databases however, the support of a co-location is uncertain; it is defined by a discrete probability distribution function (PDF).

Definition 15 (Probabilistic Support)**.** *Let* $\mathcal{DB}$ *be an uncertain co-location database and let* $X \subseteq \mathcal{F}$ *be a set of spatial features. The support of* X *is a probability density function*

$$supp(X) : I\!N_0 \rightarrow [0,1]$$

$$n \mapsto P(supp(X) = n).$$

that maps each non-negative integer n *to the probability that the support of features* X *equals* n.

Therefore, each set of spatial features has a *frequentness probability* – the probability that it is frequent.

The number of possible worlds $|W|$ that need to be considered for the computation of $P_i(X)$ is extremely large. In fact, we have $O(2^{|T|\cdot|I|})$ possible worlds, where $|I|$ denotes the total number of items. In the following, we show how to compute $P_i(X)$ without materializing all possible worlds [36].

Lemma 1. For an uncertain transaction database T with mutually independent transactions and any $0 \leq i \leq |T|$, the support probability $P_i(X)$ can be computed as follows:

$$P_i(X) = \sum_{S\subseteq T, |S|=i} (\prod_{t\in S} P(X \subseteq t) \cdot \prod_{t\in T-S} (1 - P(X \subseteq t))) \qquad (12.1)$$

Note that the transaction subset $S \subseteq T$ contains exactly i transactions.

Proof. The transaction subset $S \subseteq T$ contains i transactions. The probability of a world w_j where all transactions in S contain X and the remaining $|T - S|$ transactions do not contain X is $P(w_j) = \prod_{t\in S} P(X \subseteq t) \cdot \prod_{t\in T-S}(1 - P(X \subseteq t))$. The sum of the probabilities according to all possible worlds fulfilling the above conditions corresponds to the equation given in Definition 15. □

Support Probability Estimation

An approximation of the probabilistic support has been proposed in [188]. Here, the idea is to approximate the probabilistic support (cf. Definition 15) by a Poisson distribution. For each set of spatial features X, the single parameter λ of the Poisson distribution $Po(\lambda)$ used to approximate the support distribution of X corresponds to the expected support of X, which can be computed analogously to solutions using expected support (cf. Subsection 12.4.2). Then, the probability that the support of X exceeds $minSup$ can be computed by evaluating the cumulative distribution function of $Po(\lambda)$:

$$P(Po(\lambda) \geq minSup) = 1 - P(Po(\lambda) < minSup) =$$
$$e^{-\lambda} \sum_{i=0}^{minSup-1} \frac{\lambda^i}{i!}.$$

12.5 Experiments

We examined our experiment on as subset of our data set discussed previously in this thesis. We use 8 Million tweets, again from our data set discussed in the pevious parts of this thesis. Our data set ranges from September 2014 to March 2015 (approx. 1954 per hour) that were geo-tagged within the county of Los Angeles, USA. We decided to use Los Angeles since the tweet density is fairly high there. We discretized time into slots of one hour. Smaller timeslots would have resulted in fewer co-locations, whereas larger values would yield less interesting results, e.g., users a and b patronized the same restaurant within the same day.

We cross-referenced this data with points of interest out of OpenStreetMap[4], out of which around 16 thousand were within the investigated region and of a fitting type (we excluded points like traffic lights or garbarge bins). We paired each of these points of interest with all observations (tweets) within their τ_d-meter neighborhood and selected those pairs of PoIs p and timeslots t that contained at least two distinct observations. Each of these 184.452 (p, t)-pairs also specifies a list of the observed users with their respective sojourn probability at p.

For evaluation we implemented an algorithm based on Apriori [14] to estimate support probabilities. Figure 12.5 shows a performance evaluation against a simple enumeration of user combinations, which becomes practically unusable after surpassing only a few observations. For an input of between 10 and 250 distinct observations, we recorded the runtime to calculate support. As the graph shows, a full enumeration exhibits a super-exponential growth after about 25 observations, while the estimation approach terminates in interactive time.

[4]http://www.openstreetmap.org/

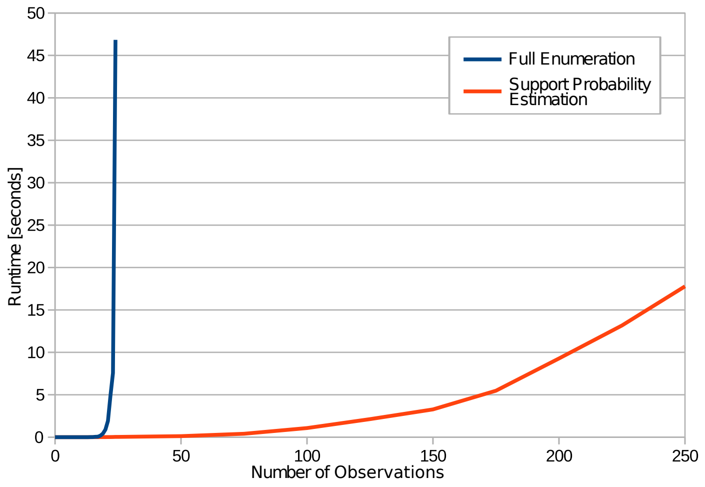


Figure 12.5: Calculation runtime comparison between enumeration and an estimation of support probability.

12.6 Conclusions

In this chapter we developed an efficient solution for finding probabilistic co-location patterns in uncertain locations, e.g., inaccurate observations derived from social media data. Our solution is mainly based on techniques used for probabilistic frequent itemset mining. In our experiments we showed that the proposed methods enable co-location mining in data sets significantly larger than possible using straightforward methods.

Chapter 13

Socio Textual Mapping of Events (Vision)

As we have discussed in this thesis so far, we are able to extract events from social media, form larger topics out of co-occurring trend terms, use location information to detect local events and predict their spreading based on dissimilation archetypes. In the following chapter we will extend our Event Detection process with a vision to describe a spatial region by the thoughts, ideas and emotions frequently and recently expressed by people in that region. For this purpose, we envision to extract features from geo-textual data, which capture not only the vocabulary, but also current topics and current general interests. We formally define the problem of drawing a socio textual map using geo-textual data and identify the necessary steps towards this vision: We represent each region as a stream of text messages such as tweets. In each region, we maintain a feature representation of text messages. We define a dissimilarity measure between such collections to assess the similarity between two regions. Using this measure, we utilize a metric clustering approach to obtain a social map of similar regions. We present a proof of concept by implementing the aforementioned steps with initial solutions. This proof of concept shows that an initial solution, which clusters the feature representations of regions, also yields clusters having regions that are spatially close. We theoretically explain this proof of concept by Tobler's first law of geography.

13.1 Introduction

Traditionally, a spatio-temporal database consists of triples (objectID, time, location), mapping objects (e.g., users) and time to a position in geo-space where the object was, is, or will be located. In recent application, this geo-information is further enriched by textual information: For example, in geo-social networks user can check-in at their current location such as a restaurant and publish a textual description of their experience at this location. Another example is Twitter, where many tweets contain a geographical tag corresponding to the geo-spatial position of the user. Loosely speaking, the textual content of a tweet contains information about *what's on the mind of a user*: For example, a tweet may describe an experience that a user wants to share, a restaurant that a user wants to recommend, an achievement that the user wants to boast about, or simply anything the user wants to say. In this chapter, we want to generalize this concept, by making the assumption that the collection of recent tweets of a region reflects *what's on the mind of a region.*

As an example, consider the topics *"Justin Bieber"* and *"Greek Bankruptcy"* and consider two geo-spatial regions, such as *Ontario, Canada* and *Germany.* It may turn out that in Ontario, one percent of all tweets contain the keyword "Justin", and five percent of all tweets contain the word "Greece". In contrast, Twitter users in Germany may use the keyword "Justin" in only 0.1 percent of their tweets, but use the keyword "Greece" in 10 percent of their tweets. Clearly, these two distributions of keywords are different. Thus, people in Ontario and people in Germany have different things that they tweet about - different things that are on their mind. We want to automatically extract a feature representation of what's on the mind of people.

In the past, such a vision of describing a region by text messages published in that region was entirely infeasible. Even in the example above, if we only have a few hundreds of tweets per day in Germany, then making significant statement about the frequency of the topic *"Justin Bieber"* is hard. Trying to make conclusions about the frequency of rare topics such as "Databases" was hard. Drawing conclusions for smaller spatial regions, such as cities or parts of cities was completely impossible. But now, both the current trends in technology such as smart phones, general mobile devices, stationary sensors and satellites as well as a new user mentality of utilizing this technology to voluntarily share information produce a huge flood of geo-textual data. Today, we have 500 million tweets per day[1] which, in addition to other sources of geo-textual data such as travel blogs and social networks, we are suddenly able to make significant conclusions about the frequency of rare terms even in small spatial regions. It's time to use this data.

In [53], Cheng et al. proposed a framework to predict a twitter users city-level based solely on words in corresponding tweets. We generalize this idea to not only determine words that classify cities, but find the latent concepts and topics that describe a generic region. Thus we extend the relationship to areas such as districts, cities, states or even artificial regions not tied to political borders (e.g. a music festival event at an off-site location).

Our vision is to describe geo-spatial regions by a representation of their thoughts. Using this representation, we want to hierarchically cluster the world in terms of what's on the mind of their people.

[1]`https://about.twitter.com/company`

We call the resulting a *socio textual map*, envisioned to be useful in a large variety of application fields:

- Research in sociology has focused on the problem of ghettos and social tensions in modern cities [87, 59, 88]. Data used in this kind of research uses Census data using "up to six race/ethnicity groups (white, black, Hispanic, Indian, Asian and other)"[88]. We claim that, especially in the 21st century, the race/ethnicity distribution of regions is *not* the sole source of social tension. Social tension may be caused simply by having different opinions and beliefs. With our solution, we can find spatial regions, on a city scale, having people with significant different interests. This may or may not be a result of ethnic differences. Our proposed approach contains much more facets of people, by directly mining the interests of the crowd.

- Our research may improve the process of geocoding of geo-textual data. Given a user who specified "London" as his location, the probabilistic distribution might be shifted towards the city of London, Ontario, Canada, if the vocabulary, topics and keywords of his tweets are more similar to regions within that area. This can be done by describing the user, who is to be geocoded, by the set of his own tweets, obtain a proper feature representation and compare this representation to candidate geo-locations.

- For targeted marketing, it may be much more interesting for a company to direct their advertisements to an area of people having a similar mindset. Even if this region covers multiple political regions. For example, an upper-class car manufacturer may be looking to direct an advertizement at a wealthy city district. However, parts of the administrative city districts may not actually wealthy, or the actual wealthy population may reach outside of the city district. With our approach, the car manufacturer can target it's advertizement at the mental cluster that is rooted in the wealthy city district.

13.2 Overview

In Section 13.3 we formalize our vision for a socio textual map, and identify the research challenges that need to be solved in this field. In Section 13.5, we implement a first solution, by solving each of the research challenges in an initial way. We show that our vision is feasible: if the necessary research steps are all solved thoroughly, then a large scale solution to map the minds of people is a vision that may become reality.

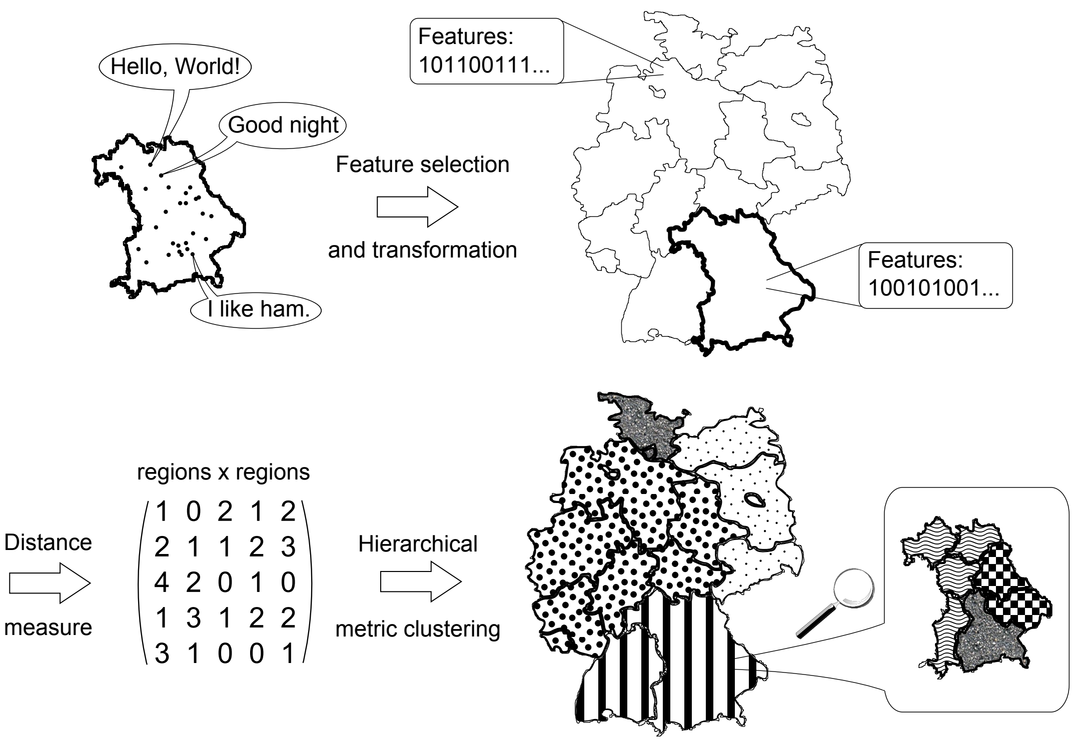


Figure 13.1: Searching in collections of multi-represented users.

13.3 Socio Textual Maps

In this section, we formally define our notion of our vision of a socio textual map. We present a theoretical foundation to prove why the concept of a socio textual map is feasible and discuss problems, open research questions and challenges. The first step requires to obtain a feature representation of a potentially large and dynamic set of textual documents.

Definition 16 (Feature Selection)**.** *Let $\mathcal{S} \in String^*$ be a set of text documents. A function $f : \mathcal{S} \mapsto R^d$ is called a d-dimensional feature representation of $\mathcal{S}$.*

The choice of function f is one of the main challenges. This function should chosen such that two of text documents $\mathcal{S}_1$ and $\mathcal{S}_2$ are similar in terms of the topics, interests and experience of these texts, if and only if $f(\mathcal{S}_1)$ is similar to $\mathcal{S}_2$. Thus, a proper feature selection method should discard terms without informative content, i.e. words that appear very frequently. A common approach is referred to as term frequency–inverse document frequency (TF-IDF) as introduced in Section 3.5 of Part I of this thesis. However, TF-IDF does not provide information about the importance of a keyword in terms of describing the mental topic of the user generating the text. This is the challenge of feature extraction for socio textual mapping.

In [61] a concept is presented that uses an entropy measure to select those features that carry the most information. Taking this concept to feature selection for geo-textual data, the idea is to select terms which are highly frequent in only a few regions, and extremely rare in others. Such local trends may be extremely useful to distinguish regions at a local scale, but may become useless for other scales and other areas. However, it seems intuitive that a proper approach should also include global trends, that most of the world has (to different degrees) on their mind. Next, we apply function f to geo-spatial regions.

Definition 17 (Feature Transformation)**.** *Let $\mathcal{W}$ denote a hierarchical partitioning of the geo-spatial region representing the surface of the earth. Each level of $\mathcal{W}$ corresponds to a geographic scale, i.e., continental level, country level, state level and city level. For each region $w \in \mathcal{W}$, the function $text(w) : \mathcal{W} \mapsto String^*$ returns a set of text documents that are associated with w.*

The first step of our workflow in Figure 13.1 illustrates this step. In the top-left of Figure 13.1, we consider a spatial region w corresponding to Bavaria, Germany. We take the set of text messages *text*(Bavaria) and apply a (in this example binary) feature transformation. The same feature transformation is performed to all other German states. In the next step, we need to assess the pair-wise dissimilarity between these regions, using standard vector distance functions. Using the resulting dissimilarity matrix, exemplarily depicted in the lower-left of Figure 13.1, we can apply a metric clustering approach to find groups of similar regions. The choice and the parameterization of this clustering approach are another challenging step. For instance, the clustering approach needs to account for different geographic scale. That is, regions on country level should allow much more freedom to be considered similar than regions on a city level.

13.4 Theoretic Foundation

There are two main theoretical reasons why finding a socio textual map is viable and feasible. The first is *the law of large numbers*, and the second is Tobler's first law of geography.

The Law of Large Numbers states that, for a random variable, the empirical probability approaches the actual probability as more trials are performed. Applied to our problem, we can treat the topic of a tweet as a random variable. For a sufficiently large number of tweets drawn from a region, the law of large number states that the fraction of tweets having a specific topic converges to the true fraction of people having this topic on their mind. Two implicit assumptions are made here. First, we assume that a the content of a tweet correlates with what on the mind of the tweeter; second, we assume that sample tweets are drawn independent without any bias. The second assumption is critical, as some people are "more vocal" than other, thus tweeting more often than others and thus, overrepresenting the topics on their minds. However, it was shown in [177] that the law of large numbers still holds as long the there is no user which *dominates* all other users in terms of tweet frequency, and as long as the error is unbiased, i.e., the frequency of tweets of a user is independent to the topics on his mind. The flood of daily tweets and other geo-textual data sets allows us to exploit the law of large number to obtain a representative sample of the minds of people in a region.

Tobler's First Law of Geography states that "everything is related to everything else, but near things are more related than distant things" [182] and is one of the key reasons why "spatial is special" [121]. It is the reason why we expect that a clustering of the minds of regions results in a clustering that is also spatially correlated. And it is the reason why we envision that we can obtain a socio textual map that captures more than lingual vocabulary, but also captures topics and trends that people think about.

13.5 Proof of Concept

For a proof of concept we use our Twitter data set as described previously in Section 3.10. On each tweet's text we applied a standard tokenizer to extract word tokens (see Section 3.6 for more details on the process of tokenization). For each geo-coordinate we use the administrative boundary lookup library, that we previously introduced in Section 6.6 in Part II of this thesis, to obtain the corresponding city. In Section 13.5.1 we present an initial solution to derive features corresponding to the relevance of terms based on their relevance to a city. In Section 13.5.2 we will present different clustering results which show that nearby cities are generally more related than cities further away from each other.

13.5.1 Feature Selection and Transformation

Each tweet t is represented as a tuple $(c_t, W_t) \in S$ where c_t denotes the city corresponding to the tweet's location and $W_t = \{w_{t,1}, ..., w_{t,n}\}$ denotes the set of extracted word tokens for the text of tweet t. After each epoch (say, 1 hour), we want to obtain a set of k most "representative" features for each city c. Using only the top-k most frequent terms would result in a sole clustering of languages, rather than a clustering of people's thoughts, because regions are usually dominated by the English language and thus, stop words like the, and, or, etc. would cause regions of actually different mind sets to be indistinguishable among themselves. Let tf_w and $\mathrm{tf}_{w,c}$ be the frequency of word token w and the frequency of w mentions within a city c.

We then define a score of a word token w as follows:

$$\text{score}(w) = \frac{P(w|c) - P(w)}{\sqrt{\text{tf}_w}} \text{tf}_{w,c}$$

where $P(w)$ denotes the probability of obtaining the word token w regardless of its location and $P(w|c)$ the probability of obtaining word token w given a city c respectively.

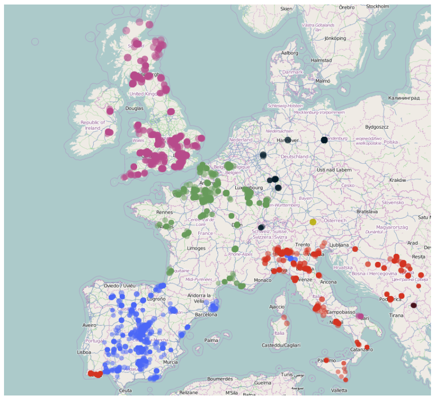
13.5.2 Clustering

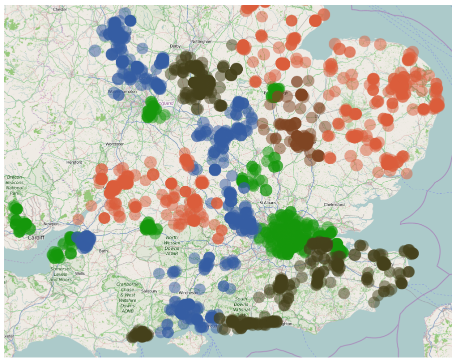
To respect the hierarchical nature of our data we utilize an agglomerative clustering approach. Therefore we first construct a similarity matrix of the size $n \times n$, where n is the number of all cities. The cell $c_{i,j}$ for the i-th and j-th city is set to the Jaccard similarity coefficient of the corresponding feature sets F_i and F_j, i.e. $c_{i,j} = \frac{|F_i \cap F_j|}{|F_i \cup F_j|}$. As shown in Figure 13.2a we are able to determine political country boundaries. Each circle represents a city having at least 150 tweets in a time-frame of 60 minutes. On a finer geographical scale, we obtain more detailed micro-clusters within countries in Europe, such as Great Britain in Figure 13.2c), and France and Spain in Figure 13.2d. On a even more detailed scale, we can find clusters within a city, such as the city of London (England) in Figure 13.2b. These different clusterings are obtained by varying the distance threshold parameter of the clustering algorithm. It is important to note that the selected features do *not* consider spatial distances. Nevertheless, the clusters that we obtain, in all settings of Figure 13.2 are, more or less, grouped in geo-space. We contribute this result to Tobler's first law of geography as presumed in Section 13.4.

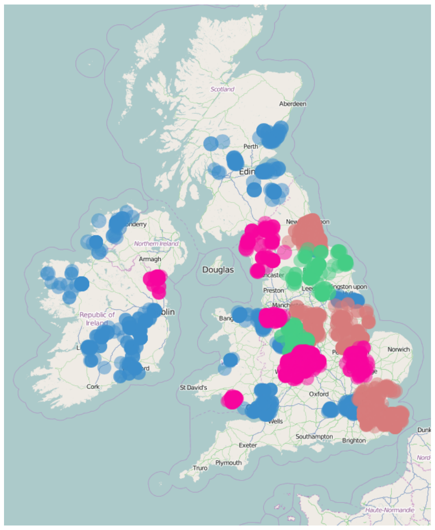
The main conclusion drawn from this evaluation is that it is possible to invert Tobler's law: We show the counter-direction that regions having similar tweets are also more like co-located in geo-space. For instance, in the experiment of Figure 13.2e we only consider tweets that are flagged to be of Spanish language. We see that cities in Spain form a cluster that, except for a few outliers, represents all of Spain and Spain only - and cities in Mexico form a cluster that, for the most part, represents all of Mexico and Mexico only. Note that this experiment also shows some cities in the United States, as these may also have more than 150 tweets in Spanish language over the considered time period. This observation is a proof of concept for the vision of socio textual mapping using user data to describe the mind of people of a region.

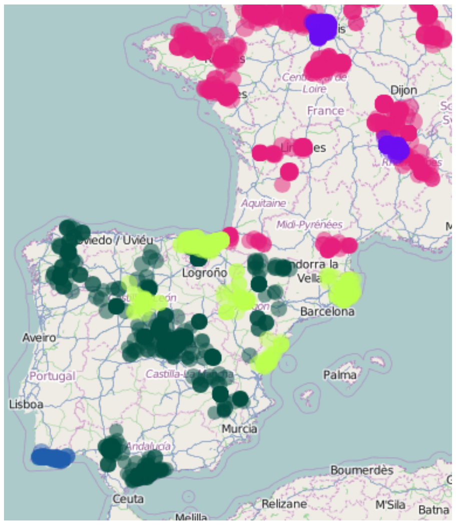
13.6 Challenges

Now that the concept is defined, theoretically founded and an empirical proof of concept is provided, we identify some of the challenges that need to be addressed by the research community for this vision to become a success.


(a) Clustering at a high level yields country boundaries.


(b) Language independent micro-cluster at city level(e.g.London).


(c) Language independent cluster results at country level


(d) Clusters within Spain and France

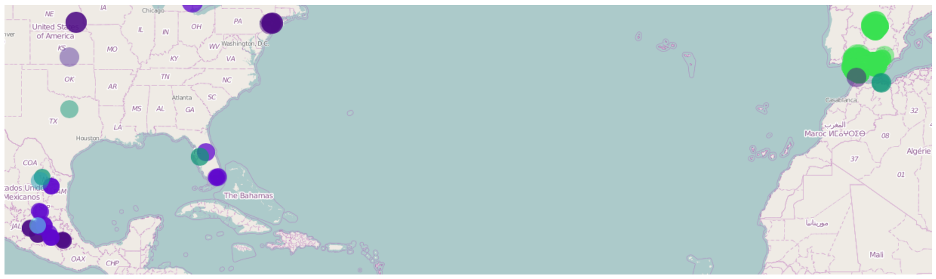

(e) Separation for same Language (Spanish) into distinct clusters.

Figure 13.2: Visual clustering results.

Feature Representation: We need to find a feature representation of tweets of a region that capture the mind of the people of that region, rather than their vocabulary. In our proof of concept, we used the most frequently used terms in a region. Clearly, any region in the US or any English speaking country will frequently use *stop words* such as "the" and "and". While our proof of concept in Section 13.5 removes these stop words, it is still an open question weather the remaining keywords are representative for the interests and thoughts of users. A more desirable approach, which will allow to better reflect the mentality of a user, rather than his vocabulary, one could look into solutions for temporal-textual trend mining as presented in [42, 164]. This way we could directly count the mentions of topics that are, globally, on the mind of people.

Distance Measure: Given occurrences of appropriate set of keywords, or any other means of representing the mind of region, we need adequate solutions to measure the similarity between regions. In our experiments, we simply used a Jaccard index to measure similarity between the selected features sets of a region (as discussed in Section 13.5.1). We need a distance measure which also takes into account the geographic scale of the considered regions: Two small parts of a city should penalize minor difference much more drastically. Also, a good distance measure may also consider the spatial distance of the regions, thus incorporating Tobler's first law of geography in the distance function.

Metric Clustering: We use an agglomerative single-link clustering, i.e., a single-link clustering that returns a dendrogram of different clusterings for different single-link-distance parameter values. This result allowed us to manually pick single-link distance thresholds for different geographic scale. For example, we manually picked a proper single-link-distance value to obtain a good clustering on country-level, and a different value for a good clustering on city-level. While our proof of concept showed that this proper parameter choice led to solid results which were highly correlated to political (and non-necessarily lingual) borders. Yet, an approach is needed to automatically adapt its parameters for different scale-levels. Also, it would be great to compare different levels, such as large cities and small countries.

Other Data Types: For our socio textual maps, we exclusively used geo-textual data to describe the mentality of a spatial region. We made this choice since large sets of geo-tagged and user-tagged are available publicly. But this is not the end of the vision. Other data types can be used to estimate the mindset of a region. For instance, time-series of activity of users, content of published multi-media data, attributes of users of a region (e.g. age, nationality, etc.). We envision a feature representation to capture all social information available, in order to plug this feature representation into the framework of Figure 13.1.

Independence of Languages: We see that lingual borders also imply mental borders. Ultimately, we envision a solution that is completely independent of languages. Thus, we want a region in Japan to be able to cluster with a region in Spain, if and only if both regions have the same topics in mind, but use different vocabulary (and letters) to describe the thoughts on their mind. Thus, we envision feature representation that are not merely based on vocabulary, but more precisely capture the mentality of people.

Part IV

Conclusion

In this thesis, we have discussed several solutions to handle the ever increasing amount of live textual data that arises from social media, blogs and daily news. The core algorithm therefore is our efficient and scalable Event Detection discussed in Part II (basic concepts were introduced in Part I).

User generated textual social media data, and in particular tweets, are complex and unstructured. They consist of spelling errors, irony, misunderstandings, abbreviations, locations, images and hyperlinks. In addition to that, computer generated content, produced from advertising campaigns, teen idol mass followings or spammers, makes this data even more difficult to overlook and analyse. Thus, an universally accepted gold standard of all recent trends and events does not exist yet; even humans often disagree on whether or not a recent happening is an event or not. Due to this fuzziness, we define an event based on quantitative data and evaluate it with well understood statistics: We call an observation an *event*, as soon as its actual occurrence frequency X exceeds its expected frequency $\mathbb{E}[X]$ by τ standard deviations σ such that $X > \tau \cdot \sigma(X) + \mathbb{E}[X]$. Thereby, we have modeled the expected frequency $\mathbb{E}[X]$ with an exponentially weighted moving average $EWMA$. This model allows intuitive adjustments of the $EWMA$ learning rate (how important is historical data for the prediction of the expected frequency) as well as adjustments for the sensitivity threshold τ (how significant our deviation needs to be). Due to this control over sensitivity and the amount of reported event instances we enable professional corporate use cases (e.g. trend- and technology-scouts) as well as casual use cases to just consume breaking news.

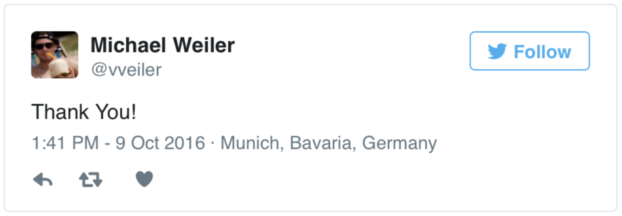
In our Event Detection we go beyond tracking single words from data sets with heavy restrictions, such as only hashtags or only small fractions of the data that includes user-predefined keywords: Our Event Detection does *not* have such restrictions and tracks *all*: single words, word co-occurrences (word pairs that were mentioned together in the same tweet or article), location data (overlapping grids and hierarchical administrative boundaries) as well as location-word combinations to discover local significant events. This allows us to report significant local events as well as significant global events at the same time while accounting for differences in adoption of social media across cultures (e.g. London produces more tweets than all of Germany).

To produce a meaningful reporting, we aggregate event keywords to larger topics by hierarchical clustering of our event co-occurrences. All this needs to be done in real-time with a minimal delay to be able to report events as they happen. Because we do not have restrictions and also track location as well as co-occurrences, naive statistical tracking becomes infeasible. We therefore proposed an efficient hashing-based probabilistic approach that is easily able to handle the full Twitter stream on a standard computer without special parallelization. If still more performance is needed, our algorithm is ready to be implemented for distributed cluster setup. Our approach takes advantage of the long-tail distribution of textual data which means that the majority of words is rare. As we are looking for significant events we are not interested in rare observations.

In the case of Twitter, we could observe that $\approx 99\%$ of the words typically appear less then 5 times within 1 hour windows. As we neither discard words nor restrict our data set beforehand, we enable a wide variety of algorithms *after* events have been efficiently detected and that otherwise would not be feasible on the raw text data. We discussed examples of such algorithms in Part III of this thesis: We showed how we are able to identify trend archetypes based on their corresponding dissemination patterns, use Geo-social co-location mining to find social groups that are frequently found at the same location or use events to obtain a social map of regions that share similar "thoughts" of users expressed by their tweets.

Acknowledgements

This thesis would not have been possible without support of my colleagues and co-authors of the publications included in this work. First of all, I would like to thank my thesis supervisor, **Prof. Dr. Hans-Peter Kriegel**, for his great support and his engagement throughout my employment as a scientific assistant at Ludwig-Maximilians-Universität München. I would like to thank him that he organized the industry funding and at the same time enabled an unconstrained working atmosphere so that I could freely work on research topics that interested me the most. Secondly, I would like to thank **Prof. Dr. Michael Gertz**, who agreed to review my thesis despite his high workload. Next, I want to thank **Dr. Erich Schubert** for his outstanding support and guidance. The brainstorming sessions with him inspired me and pushed my research to the limits. My fist publication with him did get accepted at KDD'14 and paved my way towards the Event Detection research topic. Without him, this publication would most likely not have been accepted there. Furthermore, I would like to thank **Dr. Arthur Zimek** for his great work regarding Outlier Detection that was one important part of the final Event Detection algorithm. Additionally, I want to thank **Dr. Andreas Züfle** and **Dr. Tobias Emrich**. Both provided great insights and guidance regarding the research community. I had a lot of fun with them developing great ideas and publications until late in the night, just before the deadline. I also want to thank **Dr. Johannes Niedermayer, Prof. Dr. Matthias Schubert** and **Klaus Arthur Schmid** for the valuable discussions we had. I want to thank PhD candidate fellow **Markus Mauder** in particular as he motivated me towards the end of my research career and always had time when I needed to talk about my professional career and job opportunities and rejections. I want to thank **Franz Krojer** and **Susanne Grienberger** for making university life possible with technical and administrative support. Further, I want to thank **all co-authors** of my publications for the great collaboration. Last but not least, I would like thank **Janina Rothfuss** for supporting me particularly in the sad moments of rejected publications. Thank you all! I had a great time!


Michael Weiler
@vveiler

Follow

Thank You!

1:41 PM - 9 Oct 2016 · Munich, Bavaria, Germany

Bibliography

[1] Data never sleeps. `https://www.domo.com/blog/2016/06/data-never-sleeps-4-0`. Accessed: 2016-10-09.

[2] What happens in one internet minute. `http://www.excelacom.com/resources/blog/2016-update-what-happens-in-one-internet-minute`. Accessed: 2016-10-09.

[3] Abdelhaq, H., Gertz, M., and Sengstock, C. Spatio-temporal characteristics of bursty words in Twitter streams. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM GIS), Orlando, FL* (2013), pp. 194–203.

[4] Abdelhaq, H., Sengstock, C., and Gertz, M. EvenTweet: Online localized event detection from Twitter. *Proceedings of the VLDB Endowment 6*, 12 (2013), 1326–1329.

[5] Abe, N., Zadrozny, B., and Langford, J. Outlier detection by active learning. In *Proceedings of the 12th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Philadelphia, PA* (2006), pp. 504–509.

[6] Achrekar, H., Gandhe, A., Lazarus, R., Yu, S.-H., and Liu, B. Predicting flu trends using Twitter data. In *Proc. INFOCOMW* (2011).

[7] Achtert, E., Hettab, A., Kriegel, H.-P., Schubert, E., and Zimek, A. Spatial outlier detection: Data, algorithms, visualizations. In *Proceedings of the 12th International Symposium on Spatial and Temporal Databases (SSTD), Minneapolis, MN* (2011), pp. 512–516.

[8] Achtert, E., Kriegel, H.-P., Schubert, E., and Zimek, A. Interactive data mining with 3D-Parallel-Coordinate-Trees. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), New York City, NY* (2013), pp. 1009–1012.

[9] Aggarwal, C., Li, Y., Wang, J., and Wang, J. Frequent pattern mining with uncertain data. In *Proceedings of the 15th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Paris, France* (2009), pp. 29–38.

[10] Aggarwal, C. C. A framework for diagnosing changes in evolving data streams. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), San Diego, CA* (2003), pp. 575–586.

[11] Aggarwal, C. C. *Outlier Analysis.* Springer, 2013.

[12] Aggarwal, C. C., Han, J., Wang, J., and Yu, P. S. A framework for clustering evolving data streams. In *Proceedings of the 29th International Conference on Very Large Data Bases (VLDB), Berlin, Germany* (2003), pp. 81–92.

[13] Aggarwal, C. C., and Yu, P. S. Outlier detection with uncertain data. In *Proceedings of the 8th SIAM International Conference on Data Mining (SDM), Atlanta, GA* (2008), pp. 483–493.

[14] Agrawal, R., Imieliński, T., and Swami, A. Mining association rules between sets of items in large databases. In *ACM SIGMOD Record* (1993), vol. 22, ACM, pp. 207–216.

[15] Akoglu, L., and Faloutsos, C. Anomaly, event, and fraud detection in large network datasets. In *Proceedings of the 6th ACM International Conference on Web Search and Data Mining (WSDM), Rome, Italy* (2013), pp. 773–774.

[16] Akoglu, L., McGlohon, M., and Faloutsos, C. OddBall: spotting anomalies in weighted graphs. In *Proceedings of the 14th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Hyderabad, India* (2010), pp. 410–421.

[17] Akoglu, L., Tong, H., and Koutra, D. Graph-based anomaly detection and description: A survey. *Data Mining and Knowledge Discovery 29*, 3 (2015), 626–688.

[18] Akoglu, L., Tong, H., Vreeken, J., and Faloutsos, C. Fast and reliable anomaly detection in categorical data. In *Proceedings of the 21st ACM Conference on Information and Knowledge Management (CIKM), Maui, HI* (2012), pp. 415–424.

[19] Allan, J., Lavrenko, V., and Jin, H. First story detection in tdt is hard. In *Proceedings of the ninth international conference on Information and knowledge management* (2000), ACM, pp. 374–381.

[20] Allan, J., Lavrenko, V., Malin, D., and Swan, R. Detections, bounds, and timelines: UMass and TDT-3. In *Proceedings of Topic Detection and Tracking (TDT–3)* (2000), pp. 167–174.

[21] Alvanaki, F., Michel, S., Ramamritham, K., and Weikum, G. See what's enBlogue: real-time emergent topic identification in social

media. In *Proceedings of the 15th International Conference on Extending Database Technology (EDBT), Berlin, Germany* (2012), pp. 336–347.

[22] Angiulli, F., and Fassetti, F. DOLPHIN: an efficient algorithm for mining distance-based outliers in very large datasets. *ACM Transactions on Knowledge Discovery from Data (TKDD) 3*, 1 (2009), 4:1–57.

[23] Angiulli, F., and Fassetti, F. Distance-based outlier queries in data streams: the novel task and algorithms. *Data Mining and Knowledge Discovery 20*, 2 (2010), 290–324.

[24] Angiulli, F., and Pizzuti, C. Fast outlier detection in high dimensional spaces. In *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD), Helsinki, Finland* (2002), pp. 15–26.

[25] Anselin, L. Local indicators of spatial association–LISA. *Geographical Analysis 27*, 2 (1995), 93–115.

[26] Appice, A., Ciampi, A., and Malerba, D. Summarizing numeric spatial data streams by trend cluster discovery. *Data Mining and Knowledge Discovery 29*, 1 (2015), 84–136.

[27] Aramaki, E., Maskawa, S., and Morita, M. Twitter catches the flu: detecting influenza epidemics using Twitter. In *Proc. EMNLP* (2011).

[28] Assent, I., Kranen, P., Baldauf, C., and Seidl, T. AnyOut: Anytime outlier detection on streaming data. In *Proceedings of the 17th International Conference on Database Systems for Advanced Applications (DASFAA), Busan, South Korea* (2012), pp. 228–242.

[29] Babcock, B., Datar, M., Motwani, R., and O'Callaghan, L. Maintaining variance and k-medians over data stream windows. In *Proceedings of the 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, San Diego, CA* (2003), pp. 234–243.

[30] Babcock, B., and Olston, C. Distributed top-k monitoring. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), San Diego, CA* (2003), pp. 28–39.

[31] Bansal, N., and Koudas, N. Blogscope: a system for online analysis of high volume text streams. In *Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB), Vienna, Austria* (2007), pp. 1410–1413.

[32] Barnett, V., and Lewis, T. *Outliers in Statistical Data*, 3rd ed. John Wiley&Sons, 1994.

[33] Bay, S. D., and Schwabacher, M. Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In *Proceedings of the 9th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Washington, DC* (2003), pp. 29–38.

[34] Berge, C. *Graphs and hypergraphs*, vol. 6. Elsevier, 1976.

[35] Bernecker, T., Kriegel, H.-P., Renz, M., Verhein, F., and Züfle, A. Probabilistic frequent pattern growth for itemset mining in uncertain databases. In *Scientific and Statistical Database Management*. 2012, pp. 38–55.

[36] Bernecker, T., Kriegel, H.-P., Renz, M., Verhein, F., and Züfle, A. Probabilistic frequent itemset mining in uncertain databases. In *Proceedings of the 15th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Paris, France* (2009), pp. 119–128.

[37] Bernecker, T., Kriegel, H.-P., Renz, M., and Züfle, A. Hot item detection in uncertain data. In *Proceedings of the 13th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Bangkok, Thailand* (2009), pp. 673–680.

[38] Bloom, B. H. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM 13*, 7 (1970), 422–426.

[39] Breunig, M. M., Kriegel, H.-P., Ng, R., and Sander, J. LOF: Identifying density-based local outliers. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), Dallas, TX* (2000), pp. 93–104.

[40] Broder, A. Z. On the resemblance and containment of documents. In *Proceedings of the Compression and Complexity of Sequences 1997* (1997), pp. 21–29.

[41] Budak, C., Georgiou, T., Agrawal, D., and El Abbadi, A. GeoScope: Online detection of geo-correlated information trends in social networks. *Proceedings of the VLDB Endowment 7*, 4 (2013), 229–240.

[42] Budak, C., Georgiou, T., Agrawal, D., and El Abbadi, A. Geoscope: Online detection of geo-correlated information trends in social networks. *Proceedings of the VLDB Endowment 7*, 4 (2013).

[43] Calders, T., Garboni, C., and Goethals, B. Approximation of frequentness probability of itemsets in uncertain data. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on* (2010), IEEE, pp. 749–754.

[44] Campos, G. O., Zimek, A., Sander, J., Campello, R. J. G. B., Micenková, B., Schubert, E., Assent, I., and Houle, M. E. On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery* (2015).

[45] Carroll, J. D., and Chang, J.-J. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika 35*, 3 (1970), 283–319.

[46] Cataldi, M., Caro, L. D., and Schifanella, C. Emerging topic detection on Twitter based on temporal and social terms evaluation. In *Proceedings of the 10th International Workshop on Multimedia Data Mining (MDM/KDD), Las Vegas, NV* (2010), p. 4.

[47] Chan, T. M. Approximate nearest neighbor queries revisited. *Discrete & Computational Geometry 20*, 3 (1998), 359–373.

[48] Chandola, V., Banerjee, A., and Kumar, V. Anomaly detection: A survey. *ACM Computing Surveys 41*, 3 (2009), Article 15, 1–58.

[49] Chang, J., Boyd-Graber, J. L., Gerrish, S., Wang, C., and Blei, D. M. Reading tea leaves: How humans interpret topic models. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems (NIPS), Vancouver, BC* (2009), vol. 22, pp. 288–296.

[50] Chawla, S., and Sun, P. SLOM: A new measure for local spatial outliers. *Knowledge and Information Systems (KAIS) 9*, 4 (2006), 412–429.

[51] Chen, C. CiteSpace II: Detecting and visualizing emerging trends and transient patterns in scientific literature. *Journal of the American Society for Information Science and Technology 57*, 3 (2006), 359–377.

[52] Chen, F., Lu, C.-T., and Boedihardjo, A. P. GLS-SOD: A generalized local statistical approach for spatial outlier detection. In *Proceedings of the 16th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Washington, DC* (2010), pp. 1069–1078.

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

[54] Cho, E., Myers, S. A., and Leskovec, J. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining* (2011), ACM, pp. 1082–1090.

[55] Chow, C.-Y., Mokbel, M. F., and Aref, W. G. Casper*: Query processing for location services without compromising privacy. *ACM TODS 34*, 4 (2009), 24.

[56] Chui, C.-K., and Kao, B. A decremental approach for mining frequent itemsets from uncertain data. In *Proceedings of the 12th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Osaka, Japan* (2008), pp. 64–75.

[57] Chui, C.-K., Kao, B., and Hung, E. Mining frequent itemsets from uncertain data. In *Proceedings of the 11th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Nanjing, China* (2007), pp. 47–58.

[58] Cormode, G., and Muthukrishnan, S. An improved data stream summary: the count-min sketch and its applications. *J. Algorithms 55*, 1 (2005), 58–75.

[59] Cutler, D. M., and Glaeser, E. L. Are ghettos good or bad? Tech. rep., National Bureau of Economic Research, 1995.

[60] Dang, X. H., Assent, I., Ng, R. T., Zimek, A., and Schubert, E. Discriminative features for identifying and interpreting outliers. In *Proceedings of the 30th International Conference on Data Engineering (ICDE), Chicago, IL* (2014), pp. 88–99.

[61] Dash, M., and Koot, P. W. Feature selection for clustering. In *Encyclopedia of database systems*. 2009, pp. 1119–1125.

[62] Datar, M., Gionis, A., Indyk, P., and Motwani, R. Maintaining stream statistics over sliding windows. *SIAM Journal on Computing 31*, 6 (2002), 1794–1813.

[63] Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., and Harshman, R. A. Indexing by latent semantic analysis. *Journal of the American Society for Information Science 41*, 6 (1990), 391–407.

[64] Deligiannakis, A., Kotidis, Y., and Roussopoulos, N. Compressing historical information in sensor networks. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), Paris, France* (2004), pp. 527–538.

[65] Earle, P. S., Bowden, D. C., and Guy, M. Twitter earthquake detection: earthquake monitoring in a social world. *Annals of Geophysics 54*, 6 (2012).

[66] Emmott, A. F., Das, S., Dietterich, T., Fern, A., and Wong, W.-K. Systematic construction of anomaly detection benchmarks from real data. In *Workshop on Outlier Detection and Description, held in conjunction with the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, IL* (2013), pp. 16–21.

[67] Emrich, T., Hofer, O., Kolb, A., Niedermayer, J., Seiler, N., and Weiler, M. Video route. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems* (2015), ACM, p. 95.

[68] Finch, T. Incremental calculation of weighted mean and variance. Tech. rep., University of Cambridge, 2009.

[69] Färber, I., Günnemann, S., Kriegel, H.-P., Kröger, P., Müller, E., Schubert, E., Seidl, T., and Zimek, A. On using class-labels in evaluation of clusterings. In *MultiClust: 1st International Workshop on Discovering, Summarizing and Using Multiple Clusterings Held in Conjunction with KDD 2010, Washington, DC* (2010).

[70] Gao, J., Liang, F., Fan, W., Wang, C., Sun, Y., and Han, J. On community outliers and their efficient detection in information networks. In *Proceedings of the 16th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Washington, DC* (2010), pp. 813–822.

[71] Gao, J., and Tan, P.-N. Converting output scores from outlier detection algorithms into probability estimates. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM), Hong Kong, China* (2006), pp. 212–221.

[72] Georgiadis, D., Kontaki, M., Gounaris, A., Papadopoulos, A. N., Tsichlas, K., and Manolopoulos, Y. Continuous outlier detection in data streams: an extensible framework and state-of-the-art algorithms. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), New York City, NY* (2013), pp. 1061–1064.

[73] Graf, F., Kriegel, H.-P., and Weiler, M. Robust segmentation of relevant regions in low depth of field images. In *Proceedings of the IEEE International Conference on Image Processing (ICIP), Brussels, Belgium* (Sept. 2011).

[74] Graf, F., Kriegel, H.-P., and Weiler, M. Robust image segmentation in low depth of field images. *Computing Research Repository* (2013).

[75] Grubbs, F. E. Procedures for detecting outlying observations in samples. *Technometrics 11*, 1 (1969), 1–21.

[76] Guzman, J., and Poblete, B. On-line relevant anomaly detection in the Twitter stream: an efficient bursty keyword detection model. In *Workshop on Outlier Detection and Description, held in conjunction with the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, IL* (2013), pp. 31–39.

[77] Hadi, A. S., Rahmatullah Imon, A. H. M., and Werner, M. Detection of outliers. *Wiley Interdisciplinary Reviews: Computational Statistics 1*, 1 (2009), 57–70.

[78] Han, J., Pei, J., and Yin, Y. Mining frequent patterns without candidate generation. *Proceedings of the ACM International Conference on Management of Data (SIGMOD), Dallas, TX 29* (2000), 1–12.

[79] Harshman, R. A. Foundations of the PARAFAC procedure: Models and conditions for an" explanatory" multi-modal factor analysis. *UCLA Working Papers in Phonetics 16*, 1 (1970), 84.

[80] Hermida, A., Lewis, S. C., and Zamith, R. Sourcing the arab spring: a case study of andy carvin's sources on twitter during the tunisian and egyptian revolutions. *Journal of Computer-Mediated Communication 19*, 3 (2014), 479–499.

[81] Huang, Y., Pei, J., and Xiong, H. Mining co-location patterns with rare events from spatial data sets. In *Geoinformatica* (2006), vol. 10, pp. 239–260.

[82] Huang, Y., Shekhar, S., and Xiong, H. Discovering co-location patterns from spatial data sets: A general approach. In *IEEE Transactions on Knowledge and Data Engineering* (2004), vol. 16, pp. 1472–1485.

[83] Huang, Y., Zhang, L., and Zhang, P. Finding sequential patterns from a massive number of spatio-temporal events. In *Proceedings of the 6th SIAM International Conference on Data Mining (SDM), Bethesda, MD* (2006).

[84] Indyk, P., and Motwani, R. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the 30th annual ACM symposium on Theory of computing (STOC), Dallas, TX* (1998), pp. 604–613.

[85] Jaccard, P. Distribution de la florine alpine dans la bassin de dranses et dans quelques regiones voisines. *Bulletin de la Societe Vaudoise des Sciences Naturelles 37* (1901), 241–272.

[86] Jagadish, H. V., Koudas, N., and Muthukrishnan, S. Mining deviants in a time series database. In *Proceedings of the 25th International Conference on Very Large Data Bases (VLDB), Edinburgh, Scotland* (1999), pp. 102–113.

[87] Jargowsky, P. A. Ghetto poverty among blacks in the 1980s. *Journal of Policy Analysis and Management 13*, 2 (1994), 288–310.

[88] Jargowsky, P. A. *Poverty and place: Ghettos, barrios, and the American city.* Russell Sage Foundation, 1997.

[89] Jin, W., Tung, A., and Han, J. Mining top-n local outliers in large databases. In *Proceedings of the 7th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), San Francisco, CA* (2001), pp. 293–298.

[90] Jin, W., Tung, A. K. H., Han, J., and Wang, W. Ranking outliers using symmetric neighborhood relationship. In *Proceedings of the 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Singapore* (2006), pp. 577–593.

[91] Keller, F., Müller, E., and Böhm, K. HiCS: high contrast subspaces for density-based outlier ranking. In *Proceedings of the 28th International Conference on Data Engineering (ICDE), Washington, DC* (2012), pp. 1037–1048.

[92] Kim, H.-G., Lee, S., and Kyeong, S. Discovering hot topics using Twitter streaming data social topic detection and geographic clustering. In *Proc. ASONAM* (2013).

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

[94] Knorr, E. M., and Ng, R. T. A unified notion of outliers: Properties and computation. In *Proceedings of the 3rd ACM International Conference on Knowledge Discovery and Data Mining (KDD), Newport Beach, CA* (1997), pp. 219–222.

[95] Kollios, G., Gunopulos, D., Koudas, N., and Berchthold, S. Efficient biased sampling for approximate clustering and outlier detection in large datasets. *IEEE Transactions on Knowledge and Data Engineering 15*, 5 (2003), 1170–1187.

[96] Kou, Y., Lu, C.-T., and Chen, D. Spatial weighted outlier detection. In *Proceedings of the 6th SIAM International Conference on Data Mining (SDM), Bethesda, MD* (2006).

[97] Kouloumpis, E., Wilson, T., and Moore, J. D. Twitter sentiment analysis: The good the bad and the omg! *Icwsm 11* (2011), 538–541.

[98] Kozawa, Y., Amagasa, T., and Kitagawa, H. Gpu acceleration of probabilistic frequent itemset mining from uncertain databases. In *Proceedings of the 21st ACM international conference on Information and knowledge management* (2012), pp. 892–901.

[99] Kranen, P., Kremer, H., Jansen, T., Seidl, T., Bifet, A., Holmes, G., Pfahringer, B., and Read, J. Stream data mining using the MOA framework. In *Proceedings of the 17th International Conference on Database Systems for Advanced Applications (DASFAA), Busan, South Korea* (2012), pp. 309–313.

[100] Kriegel, H.-P., Kröger, P., Schubert, E., and Zimek, A. LoOP: local outlier probabilities. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM), Hong Kong, China* (2009), pp. 1649–1652.

[101] Kriegel, H.-P., Kröger, P., Schubert, E., and Zimek, A. Outlier detection in axis-parallel subspaces of high dimensional data. In *Proceedings of the 13th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Bangkok, Thailand* (2009), pp. 831–838.

[102] Kriegel, H.-P., Kröger, P., Schubert, E., and Zimek, A. Interpreting and unifying outlier scores. In *Proceedings of the 11th SIAM International Conference on Data Mining (SDM), Mesa, AZ* (2011), pp. 13–24.

[103] Kriegel, H.-P., Kröger, P., Schubert, E., and Zimek, A. Outlier detection in arbitrarily oriented subspaces. In *Proceedings of the 12th IEEE International Conference on Data Mining (ICDM), Brussels, Belgium* (2012), pp. 379–388.

[104] Kriegel, H.-P., Schubert, M., and Zimek, A. Angle-based outlier detection in high-dimensional data. In *Proceedings of the 14th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Las Vegas, NV* (2008), pp. 444–452.

[105] Kuncheva, L. I. Change detection in streaming multivariate data using likelihood detectors. *IEEE Transactions on Knowledge and Data Engineering 25*, 5 (2013), 1175–1180.

[106] Lampos, V., De Bie, T., and Cristianini, N. Flu detector-tracking epidemics on Twitter. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD), Barcelona, Spain*. 2010, pp. 599–602.

[107] Laney, D. 3d data management: Controlling data volume, velocity and variety. *META Group Research Note 6* (2001), 70.

[108] Laniado, D., and Mika, P. Making sense of Twitter. In *The Semantic Web–ISWC 2010*. Springer, 2010, pp. 470–485.

[109] Lappas, T., Vieira, M. R., Gunopulos, D., and Tsotras, V. J. On the spatiotemporal burstiness of terms. *Proceedings of the VLDB Endowment 5*, 9 (2012), 836–847.

[110] Lappas, T., Vieira, M. R., Gunopulos, D., and Tsotras, V. J. On the spatiotemporal burstiness of terms. *Proceedings of the VLDB Endowment 5*, 9 (2012), 836–847.

[111] Lavin, A., and Ahmad, S. Evaluating real-time anomaly detection algorithms–the numenta anomaly benchmark. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)* (2015), IEEE, pp. 38–44.

[112] Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A., and Srivastava, J. A comparative study of anomaly detection schemes in network intrusion detection. In *Proceedings of the 3rd SIAM International Conference on Data Mining (SDM), San Francisco, CA* (2003).

[113] Lazarevic, A., and Kumar, V. Feature bagging for outlier detection. In *Proceedings of the 11th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Chicago, IL* (2005), pp. 157–166.

[114] Lee, J.-G., Han, J., and Li, X. Trajectory outlier detection: A partition-and-detect framework. In *Proceedings of the 24th International Conference on Data Engineering (ICDE), Cancun, Mexico* (2008), pp. 140–149.

[115] Lee, R., and Sumiya, K. Measuring geographical regularities of crowd behaviors for Twitter-based geo-social event detection. In *Proc. LBSN* (2010).

[116] Leskovec, J., Backstrom, L., and Kleinberg, J. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Paris, France* (2009), pp. 497–506.

[117] Leung, C. K.-S., Carmichael, C. L., and Hao, B. Efficient mining of frequent patterns from uncertain data. In *ICDMW '07: Proceedings of the Seventh IEEE International Conference on Data Mining Workshops* (2007), pp. 489–494.

[118] Li, J., Saha, B., and Deshpande, A. A unified approach to ranking in probabilistic databases. *Proceedings of the VLDB Endowment 2*, 1 (2009), 502–513.

[119] Li, J., Saha, B., and Deshpande, A. A unified approach to ranking in probabilistic databases. *VLDB Journal 20*, 2 (2011), 249–275.

[120] Li, J., Zou, Z., and Gao, H. Mining frequent subgraphs over uncertain graph databases under probabilistic semantics. *The VLDB Journal 21* (2012), 753–777.

[121] Li, T. J.-J., Sen, S., and Hecht, B. Leveraging advances in natural language processing to better understand tobler's first law of geography. In *ACM SIGSPATIAL* (2014), ACM, pp. 513–516.

[122] Li, W. Random texts exhibit zipf's-law-like word frequency distribution. *IEEE Transactions on information theory 38*, 6 (1992), 1842–1845.

[123] Li, W., Eickhoff, C., and de Vries, A. P. Geo-spatial domain expertise in microblogs. In *Advances in Information Retrieval - Proceedings of the 36th European Conference on IR Research (ECIR), Amsterdam, Netherlands* (2014), pp. 487–492.

[124] Liu, F. T., Ting, K. M., and Zhou, Z.-H. Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD) 6*, 1 (2012), 3:1–39.

[125] Liu, X., Lu, C.-T., and Chen, F. Spatial outlier detection: Random walk based approaches. In *Proceedings of the 18th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM GIS), San Jose, CA* (2010), pp. 370–379.

[126] Lu, C.-T., Chen, D., and Kou, Y. Algorithms for spatial outlier detection. In *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM), Melbourne, FL* (2003), pp. 597–600.

[127] Luhn, H. P. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of research and development 1*, 4 (1957), 309–317.

[128] Mamoulis, N. Co-location pattern. In *Encyclopedia of GIS*. 2008, p. 98.

[129] Mamoulis, N. Co-location patterns, algorithms. In *Encyclopedia of GIS*. 2008, pp. 103–107.

[130] Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., and Byers, A. H. Big data: The next frontier for innovation, competition, and productivity.

[131] Mateo, J., and Laguna, P. Analysis of heart rate variability in the presence of ectopic beats using the heart timing signal. *IEEE Transactions on Biomedical Engineering 50*, 3 (2003), 334–343.

[132] Mathioudakis, M., and Koudas, N. Twittermonitor: trend detection over the Twitter stream. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), Indianapolis, IN* (2010), pp. 1155–1158.

[133] McHugh, J. Testing intrusion detection systems: A critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory. *ACM Transactions on Information and System Security 3*, 4 (2000), 262–294.

[134] Mei, Q., and Zhai, C. Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining* (2005), ACM, pp. 198–207.

[135] Micenková, B., Ng, R. T., Dang, X. H., and Assent, I. Explaining outliers by subspace separability. In *Proceedings of the 13th IEEE International Conference on Data Mining (ICDM), Dallas, TX* (2013), pp. 518–527.

[136] Mueen, A., and Keogh, E. Online discovery and maintenance of time series motifs. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* (2010), ACM, pp. 1089–1098.

[137] Müller, E., Assent, I., Iglesias, P., Mülle, Y., and Böhm, K. Outlier ranking via subspace analysis in multiple views of the data. In *Proceedings of the 12th IEEE International Conference on Data Mining (ICDM), Brussels, Belgium* (2012), pp. 529–538.

[138] Müller, E., Assent, I., Steinhausen, U., and Seidl, T. OutRank: ranking outliers in high dimensional data. In *Proceedings of the 24th International Conference on Data Engineering (ICDE) Workshop on Ranking in Databases (DBRank), Cancun, Mexico* (2008), pp. 600–603.

[139] Müller, E., Schiffer, M., and Seidl, T. Adaptive outlierness for subspace outlier ranking. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management (CIKM), Toronto, ON, Canada* (2010), pp. 1629–1632.

[140] Müller, E., Schiffer, M., and Seidl, T. Statistical selection of relevant subspace projections for outlier ranking. In *Proceedings of the 27th International Conference on Data Engineering (ICDE), Hannover, Germany* (2011), pp. 434–445.

[141] Naaman, M., Becker, H., and Gravano, L. Hip and trendy: Characterizing emerging trends on Twitter. *Journal of the American Society for Information Science and Technology 62*, 5 (2011), 902–918.

[142] Nguyen, H. V., Ang, H. H., and Gopalkrishnan, V. Mining outliers with ensemble of heterogeneous detectors on random subspaces. In *Proceedings of the 15th International Conference on Database Systems for Advanced Applications (DASFAA), Tsukuba, Japan* (2010), pp. 368–383.

[143] Nguyen, H. V., Gopalkrishnan, V., and Assent, I. An unbiased distance-based outlier detection approach for high-dimensional data. In *Proceedings of the 16th International Conference on Database Systems for Advanced Applications (DASFAA), Hong Kong, China* (2011), pp. 138–152.

[144] Niedermayer, J., Züfle, A., Emrich, T., Renz, M., Mamoulis, N., Chen, L., and Kriegel, H. Probabilistic nearest neighbor queries on uncertain moving object trajectories. *PVLDB 7*, 3 (2013), 205–216.

[145] Orair, G. H., Teixeira, C., Wang, Y., Meira Jr., W., and Parthasarathy, S. Distance-based outlier detection: Consolidation and renewed bearing. *Proceedings of the VLDB Endowment 3*, 2 (2010), 1469–1480.

[146] Petrović, S., Osborne, M., and Lavrenko, V. Streaming first story detection with application to Twitter. In *Human Language Technologies* (2010), pp. 181–189.

[147] Pham, N., and Pagh, R. A near-linear time approximation algorithm for angle-based outlier detection in high-dimensional data. In *Proceedings of the 18th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Beijing, China* (2012), pp. 877–885.

[148] Platakis, M., Kotsakos, D., and Gunopulos, D. Searching for events in the blogosphere. In *Proceedings of the 18th International Conference on World Wide Web (WWW), Madrid, Spain* (2009), pp. 1225–1226.

[149] Pokrajac, D., Lazarevic, A., and Latecki, L. J. Incremental local outlier detection for data streams. In *Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining (CIDM), Honolulu, HI* (2007), pp. 504–515.

[150] Press, G. A very short history of big data. *Forbes Tech Magazine, May 9* (2013).

[151] Ramaswamy, S., Rastogi, R., and Shim, K. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), Dallas, TX* (2000), pp. 427–438.

[152] Rider, F., et al. scholar and the future of the research library.

[153] Robertson, S. Understanding inverse document frequency: on theoretical arguments for idf. *Journal of documentation 60*, 5 (2004), 503–520.

[154] Roddick, J. F., and Spiliopoulou, M. A bibliography of temporal, spatial and spatio-temporal data mining research. *ACM SIGKDD Explorations 1*, 1 (1999), 34–38.

[155] Rupert Jr, G., et al. *Simultaneous statistical inference*. Springer Science & Business Media, 2012.

[156] Sadik, M. S., and Gruenwald, L. Research issues in outlier detection for data streams. *ACM SIGKDD Explorations 15*, 1 (2013), 33–40.

[157] Sakaki, T., Okazaki, M., and Matsuo, Y. Earthquake shakes Twitter users: real-time event detection by social sensors. In *Proceedings of the 19th International Conference on World Wide Web (WWW), Raleigh, NC* (2010), pp. 851–860.

[158] Sakaki, T., Okazaki, M., and Matsuo, Y. Earthquake shakes Twitter users: real-time event detection by social sensors. In *Proceedings of the 19th International Conference on World Wide Web (WWW), Raleigh, NC* (2010).

[159] Salton, G., Wong, A., and Yang, C.-S. A vector space model for automatic indexing. *Communications of the ACM 18*, 11 (1975), 613–620.

[160] Sankaranarayanan, J., Samet, H., Teitler, B. E., Lieberman, M. D., and Sperling, J. Twitterstand: news in tweets. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems* (2009), ACM, pp. 42–51.

[161] Schmid, K. A., Frey, C., Peng, F., Weiler, M., Züfle, A., Chen, L., and Renz, M. Trendtracker: Modelling the motion of trends in space and time. In *SSTDM16* (2016).

[162] Schubert, E., Koos, A., Emrich, T., Züfle, A., Schmid, K. A., and Zimek, A. A framework for clustering uncertain data. *Proceedings of the VLDB Endowment 8*, 12 (2015), 1976–1979.

[163] Schubert, E., Weiler, M., and Kriegel, H.-P. SigniTrend: Scalable detection of emerging topics in textual streams by hashed significance thresholds. In *Proceedings of the 20th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), New York, NY* (2014), pp. 871–880.

[164] Schubert, E., Weiler, M., and Kriegel, H.-P. Signitrend: scalable detection of emerging topics in textual streams by hashed significance thresholds. In *KDD* (2014), ACM, pp. 871–880.

[165] Schubert, E., Weiler, M., and Kriegel, H.-P. Scalable detection of emerging topics and geo-spatial events in large textual streams.

[166] Schubert, E., Weiler, M., and Kriegel, H.-P. SPOTHOT: Scalable detection of geo-spatial events in large textual streams. In *Proceedings of the 28th International Conference on Scientific and Statistical Database Management (SSDBM), Budapest, Hungary* (2016).

[167] Schubert, E., Weiler, M., and Zimek, A. Outlier detection and trend detection: Two sides of the same coin. In *1st International Workshop on Event Analytics using Social Media Data at the 15th IEEE International Conference on Data Mining (ICDM)* (2015).

[168] Schubert, E., Wojdanowski, R., Zimek, A., and Kriegel, H.-P. On evaluation of outlier rankings and outlier scores. In *Proceedings of the 12th SIAM International Conference on Data Mining (SDM), Anaheim, CA* (2012), pp. 1047–1058.

[169] Schubert, E., Zimek, A., and Kriegel, H.-P. Generalized outlier detection with flexible kernel density estimates. In *Proceedings of the 14th SIAM International Conference on Data Mining (SDM), Philadelphia, PA* (2014), pp. 542–550.

[170] Schubert, E., Zimek, A., and Kriegel, H.-P. Local outlier detection reconsidered: a generalized view on locality with applications to spatial, video, and network outlier detection. *Data Mining and Knowledge Discovery 28*, 1 (2014), 190–237.

[171] Schubert, E., Zimek, A., and Kriegel, H.-P. Fast and scalable outlier detection with approximate nearest neighbor ensembles. In *Proceedings of the 20th International Conference on Database Systems for Advanced Applications (DASFAA), Hanoi, Vietnam* (2015), pp. 19–36.

[172] Shekhar, S., and Huang, Y. Co-location rules mining: A summary of results. In *Proceedings of the 7th International Symposium on Spatial and Temporal Databases (SSTD), Redondo Beach, CA* (2001), pp. 236–256.

[173] Shekhar, S., Lu, C.-T., and Zhang, P. A unified approach to detecting spatial outliers. *GeoInformatica 7*, 2 (2003), 139–166.

[174] Sim, K., Gopalkrishnan, V., Zimek, A., and Cong, G. A survey on enhanced subspace clustering. *Data Mining and Knowledge Discovery 26*, 2 (2013), 332–397.

[175] Subramaniam, S., Palpanas, T., Papadopoulos, D., Kalogeraki, V., and Gunopulos, D. Online outlier detection in sensor data using non-parametric models. In *Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB), Seoul, Korea* (2006), pp. 187–198.

[176] Sun, P., and Chawla, S. On local spatial outliers. In *Proceedings of the 4th IEEE International Conference on Data Mining (ICDM), Brighton, UK* (2004), pp. 209–216.

[177] Sveshnikov, A. A., and Gelbaum, B. R. *Problems in probability theory, mathematical statistics and theory of random functions*. Courier Corporation, 1968.

[178] Takahashi, Y., Utsuro, T., Yoshioka, M., Kando, N., Fukuhara, T., Nakagawa, H., and Kiyota, Y. Applying a burst model to detect bursty topics in a topic model. In *Advances in Natural Language Processing – Proceedings of the 8th International Conference on NLP, JapTAL 2012, Kanazawa, Japan, October* (2012), pp. 239–249.

[179] Takeuchi, J., and Yamanishi, K. A unifying framework for detecting outliers and change points from time series. *IEEE Transactions on Knowledge and Data Engineering 18*, 4 (2006), 482–492.

[180] Tang, J., Chen, Z., Fu, A. W.-C., and Cheung, D. W. Enhancing effectiveness of outlier detections for low density patterns. In *Proceedings of the 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Taipei, Taiwan* (2002), pp. 535–548.

[181] Tavallaee, M., Bagheri, E., Lu, W., and Ghorbani, A. A detailed analysis of the KDD CUP 99 data set. In *Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defence Applications 2009* (2009).

[182] Tobler, W. R. A computer movie simulating urban growth in the detroit region. *Economic geography* (1970), 234–240.

[183] Tong, Y., Chen, L., Cheng, Y., and Yu, P. S. Mining frequent itemsets over uncertain databases. 1650–1661.

[184] Tran, G. B., and Alrifai, M. Indexing and analyzing Wikipedia's current events portal, the daily news summaries by the crowd. In *Proceedings of the 23rd International Conference on World Wide Web (WWW), Seoul, Korea* (2014), pp. 511–516.

[185] Unankard, S., Li, X., and Sharaf, M. A. Emerging event detection in social networks with location sensitivity. *World Wide Web 18*, 5 (2015), 1393–1417.

[186] Vu, N. H., and Gopalkrishnan, V. Efficient pruning schemes for distance-based outlier detection. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD), Bled, Slovenia* (2009), pp. 160–175.

[187] Walther, M., and Kaisser, M. Geo-spatial event detection in the Twitter stream. In *Adv. Information Retrieval*. 2013.

[188] Wang, L., Cheng, R., Lee, S. D., and Cheung, D. Accelerating probabilistic frequent itemset mining: a model-based approach. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management (CIKM), Toronto, ON, Canada* (2010), pp. 429–438.

[189] Wang, L., Cheung, D.-L., Cheng, R., Lee, S.-D., and Yang, X. Efficient mining of frequent item sets on large uncertain databases. *Knowledge and Data Engineering, IEEE Transactions on 24*, 12 (2012), 2170–2183.

[190] Wang, L., Wu, P., and Chen, H. Finding probabilistic prevalent colocations in spatially uncertain data sets. *IEEE Transactions on Knowledge and Data Engineering 25*, 4 (2013), 790–804.

[191] Wang, X., Zhang, Y., Zhang, W., and Lin, X. Efficiently identify local frequent keyword co-occurrence patterns in geo-tagged Twitter stream. In *Proceedings of the 37th International Conference on Research and Development in Information Retrieval (SIGIR), Gold Coast, QLD, Australia* (2014), pp. 1215–1218.

[192] Watanabe, K., Ochi, M., Okabe, M., and Onai, R. Jasmine: a real-time local-event detection system based on geolocation information propagated to microblogs. In *Proceedings of the 20th ACM Conference on Information and Knowledge Management (CIKM), Glasgow, UK* (2011), pp. 2541–2544.

[193] Weiler, M., Schmid, K. A., Renz, M., and Mamoulis, N. Geo-social co-location mining. In *GeoRich: 2nd International ACM SIGMOD Workshop on Managing and Mining Enriched Geo-Spatial Data Held in Conjunction with SIGMOD 2015, Melbourne, Australia* (2015).

[194] Weiler, M., Züfle, A., Borutta, F., and Emrich, T. Socio textual mapping. In *Proceedings of the 8th ACM SIGSPATIAL International Workshop on Location-Based Social Networks* (2015), ACM, p. 6.

[195] Welford, B. P. Note on a method for calculating corrected sums of squares and products. *Technometrics 4*, 3 (1962), 419–420.

[196] Weng, J., and Lee, B.-S. Event detection in Twitter. In *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain* (2011).

[197] West, D. H. D. Updating mean and variance estimates: an improved method. *Communications of the ACM 22*, 9 (1979), 532–535.

[198] Xiong, H., Shekhar, S., Huang, Y., Kumar, V., Ma, X., and Yoo, J. S. A framework for discovering co-location patterns in data sets with extended spatial objects. In *SDM* (2004), pp. 78–89.

[199] Yamanishi, K., Takeuchi, J.-I., Williams, G., and Milne, P. Online unsupervised outlier detection using finite mixture with discounting learning algorithms. *Data Mining and Knowledge Discovery 8* (2004), 275–300.

[200] Yang, J., Zhong, N., Yao, Y., and Wang, J. Local peculiarity factor and its application in outlier detection. In *Proceedings of the 14th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Las Vegas, NV* (2008), pp. 776–784.

[201] Yang, Y., Pierce, T., and Carbonell, J. A study of retrospective and on-line event detection. In *Proceedings of the 32nd International Conference on Research and Development in Information Retrieval (SIGIR), Boston, MA* (1998), pp. 28–36.

[202] Yu, J. X., Qian, W., Lu, H., and Zhou, A. Finding centric local outliers in categorical/numerical spaces. *Knowledge and Information Systems (KAIS) 9*, 3 (2006), 309–338.

[203] Zhang, K., Hutter, M., and Jin, H. A new local distance-based outlier detection approach for scattered real-world data. In *Proceedings of the 13th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Bangkok, Thailand* (2009), pp. 813–822.

[204] Zhang, Q., Li, F., and Yi, K. Finding frequent items in probabilistic data. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), Vancouver, BC, Canada* (2008), pp. 819–832.

[205] Zhang, X., Mamoulis, N., Cheung, D. W., and Shou, Y. Fast mining of spatial collocations. In *Proceedings of the 10th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Seattle, WA* (2004), pp. 384–393.

[206] Zhou, X., and Chen, L. Event detection over twitter social media streams. *The VLDB journal 23*, 3 (2014), 381–400.

[207] Zimek, A., Campello, R. J. G. B., and Sander, J. Ensembles for unsupervised outlier detection: Challenges and research questions. *ACM SIGKDD Explorations 15*, 1 (2013), 11–22.

[208] Zimek, A., Campello, R. J. G. B., and Sander, J. Data perturbation for outlier detection ensembles. In *Proceedings of the 26th International Conference on Scientific and Statistical Database Management (SSDBM), Aalborg, Denmark* (2014), pp. 13:1–12.

[209] Zimek, A., Gaudet, M., Campello, R. J. G. B., and Sander, J. Subsampling for efficient and effective unsupervised outlier detection ensembles. In *Proceedings of the 19th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Chicago, IL* (2013), pp. 428–436.

[210] Zimek, A., Schubert, E., and Kriegel, H.-P. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining 5*, 5 (2012), 363–387.

[211] Zimek, A., and Vreeken, J. The blind men and the elephant: On meeting the problem of multiple truths in data from clustering and pattern mining perspectives. *Machine Learning 98*, 1–2 (2015), 121–155.

[212] Zimmermann, A. The data problem in data mining. *ACM SIGKDD Explorations 16*, 2 (2014), 38–45.

[213] Zou, Z., Gao, H., and Li, J. Discovering frequent subgraphs over uncertain graph databases under probabilistic semantics. In *Proceedings of the 16th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Washington, DC* (2010), pp. 633–642.

[214] Zou, Z., Li, J., Gao, H., and Zhang, S. Mining frequent subgraph patterns from uncertain graph data. *Knowledge and Data Engineering, IEEE Transactions on 22*, 9 (2010), 1203–1218.