Real-Time Event Detection using Twitter

ANDREW JAMES MCMINN

Submitted in fulfilment of the requirements for
the degree of **Doctor of Philosophy**.

School of Computing Science,
College of Science and Engineering,
University of Glasgow

March 2018

Abstract

News has transformed from something delivered to your door, and read only once per day, to a deluge of real-time updates on events as they happen around the world, delivered on a device that is kept in your pocket. Twitter is of particular interest due to its real-time nature, and has become the social network of news. Monitoring what is said on Twitter is a frequent task for anyone who requires timely access to information: journalists, traders, and the emergency services have all invested heavily in monitoring Twitter in recent years.

Given this, there is a need to develop systems that can automatically monitor Twitter and detect real-world events as they happen, and alert users to novel events. However, this is not an easy task due to the noise and volume of data that is produced from social media streams such as Twitter. Although a range of approaches have been developed, many are unevaluated, cannot scale past low volume streams, or can only detect specific types of event.

In this thesis, we develop novel approaches to event detection, and enable the evaluation and comparison of event detection approaches by creating a large-scale test collection called Events 2012, containing 120 million tweets and with relevance judgements for over 500 events. We use existing event detection approaches and Wikipedia to generate candidate events, then use crowdsourcing to gather annotations.

We propose a novel entity-based, real-time, event detection approach that we evaluate using the Events 2012 collection, and show that it outperforms existing state-of-the-art approaches to event detection whilst also being scalable. We examine and compare automated and crowdsourced evaluation methodologies for the evaluation of event detection.

Finally, we propose a Newsworthiness score that is learned in real-time for heuristically labelled data. The score is able to accurately classify individual tweets as newsworthy or noise in real-time. We adapt the score for use as a feature for event detection, and find that it can easily be used to filter out noisy clusters and improve existing event detection techniques.

Declaration

I declare that this thesis was composed entirely by myself, and that the work contained herein is my own except where explicitly stated otherwise. This work has not, in whole or in part, been submitted in any previous application for a degree.

Andrew James McMinn

24th October 2018

Contents

List of Figures

List of Tables

Publications

Some of the work presented here has previously appeared in the following publications:

- **Building a Large-Scale Corpus for Evaluating Event Detection on Twitter**
  Proceedings of the 22nd ACM international conference on Information & Knowledge Management (2013)
  Andrew James McMinn, Yashar Moshfeghi, Joemon M Jose

- **An Interactive Interface for Visualizing Events on Twitter**
  Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval (2014)
  Andrew James McMinn, Daniel Tsvetkov, Tsvetan Yordanov, Andrew Patterson, Rrobi Szk, Jesus A Rodriguez-Perez, Joemon M Jose

- **Picture the scene...;: Visually summarising social media events**
  Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (2014)
  Philip J McParlane, Andrew James McMinn, Joemon M Jose

- **Real-Time Entity-Based Event Detection for Twitter**
  International Conference of the Cross-Language Evaluation Forum for European Languages (2015)
  Andrew James McMinn, Joemon M Jose

CHAPTER 1

Introduction

News has transformed from something delivered to your door, and read only once per day, to a deluge of real-time updates on events as they happen around the world, delivered on a device that is kept in your pocket. Social media plays perhaps one of the most important roles in news delivery today, with services such as Twitter allowing users to "See what's happening in the world right now[1]". However, Twitter is not just used to consume news, but also to produce it. The first reaction of many people involved in newsworthy events – from small accidents, to headline hitting terror attacks – is to report what they are witnessing on social media.

Twitter's reputation for breaking news has grown throughout the years, transforming it from a social network for posting photos of your breakfast, to *the* social network of news. A number of high-profile news events have broken on Twitter: from the first reports of a plane landing in New York's Hudson River in 2009, to leaked reports of Osama bin Laden's death in 2011 by White House staffers. More recently, partially due to the election of Donald Trump as President of the United States, Twitter has become a broadcast tool for world leaders, with some even worrying that Twitter could be the medium through which a nuclear war is started[2] (something that, in this author's opinion, is not as far-fetched as it sounds).

Hu et al. [2012] demonstrated the effectiveness of Twitter as a medium for breaking news, and found that news of Osama bin Laden's death not only broke on Twitter, but had reached millions of people before the official announcement. Given this, it is not surprising that there is considerable interest in following world-wide events by monitoring what is said on Twitter. News organizations are employing increasingly large numbers of staff to monitor social media for breaking news, or using commercial entities such as Storyful[3] and Dataminr[4] who monitor social media on their behalf. For

[1] Twitter's slogan, as of March 2018

[2] `https://www.indy100.com/article/trump-ban-twitter-threaten-nuclear-war-big-button-north-korea-desk-8139211`

[3] `https://storyful.com/`

[4] `https://www.dataminr.com/`

almost 10 years, the Twitter account for 'Breaking News' (@breakingnews[5]), an organization which aimed to report breaking news in real-time, had over a dozen journalists working around the clock to monitor Twitter to report on breaking news stories as they happened. The account had over 9 million followers when it shut down in January 2017.

The ability to automatically detect and track on-going real-world events would clearly be useful, however it has been shown to be an extremely difficult task, even on newswire documents [Allan et al. 2000a]. Twitter poses a number of challenges over and above those found in newswire documents. Despite Twitter's reputation for news, the majority of tweets discuss trivial or mundane events. Even today, it is not uncommon for a user to only posts about the food they have eaten or the music they are listening to. The relatively low quality of tweets poses further issues. Due to the limited message length on Twitter (tweets were originally limited to 140 characters, however this was increased to 280 in 2017), spelling and grammar errors are very common, as is the use of abbreviations and acronyms. Twitter has over 500 million users who, combined, post thousands of tweets every second[6]. Approaches developed for newswire documents simply do not scale from hundreds of newswire documents per day to thousands of tweets per second, and have no way of filtering out the mundane and noisy content.

Sakaki et al. [2010] were perhaps one of the first to show how Twitter could be used to detect real-world events. They were able to use Twitter as a social-sensor to detect the size and direction of earthquakes in real-time, notifying users of incoming earthquakes much faster than even the Japan Meteorological Agency. Since then, a number of approaches have been developed that attempt to detect and track real-world events in a more general manner. Petrović et al. [2010a] used Locality Sensitive Hashing to scale a clustering approach that been found to be effective on newswire documents to Twitter scale, demonstrating that it was possible to perform real-time and generalizable event detection on Twitter, opening the door for improved approaches.

Based on this, we define a number of characteristics that we believe are important for the development of improved event detection approaches for Twitter:

- **Real-Time:** Although techniques like batch processing can, in theory, make the task of event detection easier, it also reduces usefulness. Real-time pro-

[5] https://twitter.com/BreakingNews

[6] https://blog.twitter.com/marketing/en_us/a/2015/testing-promoted-tweets-on-our-logged-out-experience.html

cessing means that event detection approaches should be able to detect events as they are still happening, and with minimal delay. Old news is not useful.

- **Generalizable:** Event detection approaches should be able to detect news of any type, and without being explicitly told what to look for. Approaches that can only detect certain types of event, such as earthquakes or sports events, are only useful in their specific domains.

- **No manual labels:** A reliance on manually labelled training data or interaction by a user would introduce a number of restrictions on the types of event that can be detected, and be vulnerable to any changes that Twitter make (such as increasing the character limit).

- **Comparable:** Without an effective method of comparing different approaches, it is impossible to say if improvements are being made. A robust and evaluation methodology and standard data set are important for progress to be made.

We first build a corpus and set of events with relevance judgements, called Events 2012, that allows us to fairly evaluate and benchmark event detection approaches for Twitter. We refine the definition of 'event' for detection on Twitter after surveying existing definitions and finding that there was a lack of agreement and consistency. We then reduce a set of over 1 billion tweets to a more manageable 120 million covering a continuous 28 day period to provide a standard set of tweets that event detection approaches can use to emulate a high-volume Twitter stream. We implement two existing event detection approaches: the Locality Sensitive Hashing (LSH) approach proposed by Petrović et al. [2010a], and the Cluster Summarization (CS) approach proposed by Aggarwal and Subbian [2012], and extract candidate events from the corpus by running both approaches over it. We perform a crowdsourced evaluation to annotate each of the candidate events as either a true event or noise, and gather relevance judgements for the tweets of each true event. We supplement these events by using Wikipedia's Current Events Portal to identify a number of events that occurred during the period covered by the event, and use another crowdsourced evaluation to gather annotations for these events. We then use a clustering approach to merge any duplicate events, and create a final set of relevance judgements that can be used to evaluate event detection approaches. We run an additional crowdsourced evaluation to examine the quality of merging approach, and find that it performs well at merging duplicate events from multiple sources.

One of the biggest challenges for real-time event detection on Twitter is efficiently clustering the huge volume of tweets in real-time. Previous work has used approaches such as Locality Sensitive Hashing (LSH) [Petrović et al. 2010a] to reduce the number of comparisons that need to be made in order to efficiently cluster tweets. We examine the structure of events at a high level, and find that named entities play a key role in describing events. We exploit this to improve clustering efficiency by partitioning tweets using the named entities they contain, and only compare tweets that have at least one overlapping named entity. This allows us to efficiently cluster tweets in real-time, then using a light-weight burst detection approach, we monitor tweet usage over time to determine when entity mentions burst, suggesting a possible event. We then use a number of heuristic features to identify interesting clusters that are likely to be related to events. We solve event fragmentation, a common problem for event detection approaches on Twitter where a single seminal event is split into multiple parts, by combining events with high entity concurrence. We evaluate our entity-based approach using the Events 2012 corpus, and find that it significant outperforms the LSH and CS approach that we use as baselines. A crowdsourced evaluation finds that the precision of our approach is more than 3 times better than suggested using an automatic evaluation approach, and we discuss some of the issues arising from the automatic evaluation of event detection approaches in situations where there are no queries and the relevance judgements are incomplete.

Generally, event detection approaches use the volume of tweets as an indication that something significant has happened, either by measuring the size of a cluster, or the frequency of a term over time. This often requires 10s of tweets before events can be detected with reasonable precision. However, the ability to automatically determine how newsworthy an individual tweet is would allow for events to be detected with fewer tweets, and much earlier. We propose a set of heuristics that can be used to automatically label tweets as High or Low Quality. We then use use these labels to feed tweets into a newsworthiness scoring model. We use this model to assign tweets with a Newsworthiness Score that can be positive or negative (Newsworthy or Noise), and can be used for both classification and scoring. We evaluate the classification accuracy and score appropriateness using relevance judgements from the Events 2012 corpus. We then propose a cluster based newsworthiness score that can be used as a feature for event detection. We evaluate the performance of our newsworthiness feature by removing clusters with low newsworthiness scores and find that it performs favourably compared to our entity based approach. A further manual evaluation finds near-perfect precision with as few as 5 tweets (0.950), and perfect precision at 50 tweets.

1.1 Research Questions

The hypothesis of this thesis is that we can automatically identify real-world events, in real time, from posts on social media, with a particular focus on Twitter. To determine if this hypothesis is true, we pose a number of key research questions that we aim to answer and that determine the scope of this thesis:

RQ1 Can we develop a methodology that allows us to build a test collection for the evaluation of event detection approaches on Twitter?

RQ2 Can entities (people, places, organizations) be used to improve real-world event detection in a streaming setting on Twitter?

RQ3 How fairly and accurately can automatic evaluation approaches evaluate event detection approaches for Twitter?

RQ4 Can we determine the newsworthiness of an individual tweet from content alone?

1.2 Contributions

In summary, the main contributions of this thesis are:

- The creation of the first large-scale corpus for the evaluation of event detection approaches for Twitter.
- A definition of 'event' for the purpose of event detection on Twitter.
- A novel, real-time event detection approach that is computationally efficient and scalable, and that outperforms existing state-of-the-art approaches.
- The first in-depth and comparable evaluation of an event detection approach for Twitter.
- A novel method of scoring tweets to determine newsworthiness, that requires no manually labelled training and that is capable of being integrated into existing event detection approaches, resulting in significant improvements to precision.

1.3 Organization of Thesis

The remainder of this thesis is organized as follows:

Chapter 2 gives background information that is needed to understand the rest of this thesis. We first describe the role of the Topic Detection and Tracking project in

developing early event detection approaches for newswire documents, and how these approaches relate to modern event detection approaches for social media. We then discuss various high-level strategies used by event detection approaches for social media, and review related work with a focus on novel approaches of event detection on Twitter.

Chapter 3 describes the creation of the Events 2012 corpus for the evaluation of event detection approaches on Twitter. We first survey existing event detection corpora for Twitter and find that none are suitable for the evaluation of event detection on Twitter. We examine various definitions of 'event' and find that no existing definition suitably captures the nuances of events and event detection on social media, so propose a new definition. We then use a combination of existing event detection approaches and Wikipedia to create a set of candidate events, which we annotate through a crowdsourced evaluation, to create the first large-scale corpus for the evaluation of event detection on Twitter. Parts of this work were first presented in McMinn et al. [2013].

Chapter 4 examines the role of named entities in events and proposes a new entity-based event detection approach. Our approach uses named entities to efficiently cluster tweets in real-time, and a light-weight burst detection technique to identify clusters that are likely to be related to real-world events. We evaluate the approach using the Events 2012 corpus created in chapter 3 and find that our approach out-performs two state-of-the-art baselines. We compare automatic evaluation approaches to a crowdsourced approach and find that although automatic evaluation of event detection on Twitter is effective, there remain a number of challenges. Parts of this work were first presented in McMinn et al. [2014] and McMinn and Jose [2015].

Chapter 5 describes a novel newsworthiness scoring approach that uses heuristics to label tweets as high or low quality, and trains a number of models in real-time that we then use to estimate newsworthiness scores for tweets. We evaluate the performance of our approach as both a classification and scoring task using the Events 2012 corpus and find that it is convincingly able to classify tweets as either newsworthy or noise. We examine how our newsworthiness score can be used as a feature for event detection, find that it can be used to significantly increase precision.

Chapter 6 summarizes the main contributions of this thesis, and examines remaining issues for event detection on Twitter, describing possible directions for future work.

CHAPTER 2

Background

In this chapter, we review the background information required to understand the remainder of this thesis. We draw on several areas of research, such as Information Retrieval (IR) and Natural Language Processing (NLP), and describe how documents are represented and compared, and detail a number of IR measures. We then give a brief overview of the Topic Detection and Tracking project, and how it related to modern approaches for event detection on Twitter. Finally, we give a survey of related work in the area of event detection Twitter and describe how it relates to the work presented in this thesis.

2.1 Information Retrieval

Fist we discuss the area of Information Retrieval, and describe some of the most common concepts that we use throughout this thesis. In a traditional IR task, the goal is to take a query describing some information need, and return a ranked list of documents that are relevant to that query. In the remainder of this section, we examine some of the basic techniques used to complete this task, and describe how they relate to event detection on Twitter.

2.1.1 Document Representation

One of the most commonly used document and query representations is the Vector Space model [Salton et al. 1975]. The Vector Space model represents both queries and documents as a vector, where each term in the document corresponds to a dimension in the vector. Terms that occur in a document will have a non-zero value in the vector, while terms that do not appear will have a value of 0. The Vector Space model is used throughout this thesis to represent documents (tweets).

TF-IDF

One of the most common ways of calculating the weight a term has in a vector is known as the Term Frequency-Inverse Document Frequency model, or TF-IDF. TF-IDF attempts to estimate the importance of a term with regards to both the collection as a whole and document itself.

The TF component is concerned with the weight of the term in relation to the document, the basic premise being that the more frequently a term occurs in a document, the more weight it should carry. The IDF component, on the other hand, is concerned with the weight of the term in relation to the corpus as a whole. The IDF component attempts to estimate the amount of information that a term carries on the basis that rare terms should be given more weight as their presence in a document is more likely to be an indication of topic, whilst common terms should be given less weight as they are less topically specific. TF-IDF is calculated as so:

$$w_{t,d} = \text{tf}_{t,d} \cdot \log \frac{|D|}{|\{d' \in D \,|\, t \in d'\}|}$$

where $\text{tf}_{t,d}$ is the term frequency of term t in document d, $|D|$ is the total number of documents in the collection, and $|\{d' \in D \,|\, t \in d'\}|$ is the number of documents contain term t. Note that for retrieval tasks on Twitter, the TF component is often ignored as tweets are very short, and generally do not contain repeated terms.

Cosine

Many IR tasks, including event detection, are concerned with the the notion of similarity. One such similarity measure is cosine similarity, which measures the cosine of the angle between two documents in vector space:

$$\cos\theta = \frac{\mathbf{d_1} \cdot \mathbf{d_2}}{\|\mathbf{d_1}\| \, \|\mathbf{d_2}\|}$$

where $\mathbf{d_1} \cdot \mathbf{d_2}$ is the dot product of term weighted vectors for the documents, and the norm for vector $\mathbf{d}$ is calculated as such:

$$\|\mathbf{d}\| = \sqrt{\sum_{i=1}^{n} d_i^2}$$

2.1.2 Evaluation Measures

Throughout this thesis, we use a number of IR standard evaluation metrics, commonly used in IR evaluations. These measures as based on the concept of relevance. In IR, relevance is determined based on some information need, usually represented as a textual query. In event detection, there is no query. Instead, relevance is defined in relation to an event, either by subjectively evaluating if a cluster of tweets meets some definition of 'event', or by matching tweets in the cluster to a pre-determined event in the relevance judgements (represented by a set of tweets). We discuss this further is chapter 4.

Recall

Recall is measured as the fraction of relevant documents that at retrieved from all possible relevant documents. Recall is give as:

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

Precision

In a traditional IR setting, precision is measured as the fraction of retrieved documents that are relevant to a given query. For the purpose of evaluating event detection approaches, the lack of a query means that we must consider relevance differently; either in terms of meeting some definition of 'event', or by matching back to some event from the relevance judgements. How this is done is examined in chapters 3 and 4.

In a standard IR setting, precision is given as:

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

F-measure

Individually, precision and recall show only one aspect of performance. Perfect recall can be achieved by returning every document in the collection, whilst perfect precision can be achieved by returning none of them.

F-measure takes both precision and recall into consideration, and combines them

into a single score. F-measure ranges between 1 at best (a perfect systems) and 0 at worst. Although it is possible to weight F-measure to prefer precision or recall, the most commonly used F-measure gives equal weight to both, and is the harmonic mean of precision and recall. This is usually called F1 score, and is used throughout this thesis:

$$F_1 = 2 \cdot \frac{1}{\frac{1}{\text{recall}} + \frac{1}{\text{precision}}} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

B Cubed

2.2 Topic Detection and Tracking

The Topic Detection and Tracking (TDT) project began in 1997, and was a body of research and an evaluation paradigm that addressed the event-based organization of broadcast news [Allan 2002a]. The motivation behind the project was a system which was capable of monitoring broadcast news and could produce an alert when a new event occurred.

The project was split into five distinct subtasks [Allan 2002a]:

- **Story Segmentation** was the task of dividing audio transcripts taken from news shows into individual stories.
- **First Story Detection** was the task of recognizing the onset of a new topic (event) in the stream of news stories.
- **Cluster Detection** was the task of grouping all stories as they arrived, based on the topics they discuss.
- **Tracking** was a task that required systems to identify news stories discussing the same topic as a set of sample stories, by monitoring the stream of news stories as they appeared.
- **Story Link Detection** was the task of deciding whether two randomly selected stories discussed the same topic (event).

With the exception of Story Segmentation, the tasks focus on various aspects of document clustering, with the main difference between each task being how it is evaluated. Of the 5 tasks, First Story Detection (FSD, although often called New Event Detection, NED) was considered the most difficult [Allan et al. 2000a] as an effective FSD system must be effective across all of the tasks to perform well.

Story Segmentation differs in nature from the others as it pertains specifically to a pre-processing step that must be performed before the other tasks can be carried out. All other tasks work with stories, however the aim of the Story Segmentation task is to detect boundaries between stories automatically using transcripts from the spoken audio of broadcast news shows. Of all the tasks of the TDT project, Story Segmentation is the least relevant to event detection on Twitter as tweets almost always discuss only a single topic due to their short length.

2.2.1 Approaches to TDT

ALGORITHM 1: A basic TDT approach, similar to that used by UMass and other TDT participants

Input: Minimum similarity threshold m

```
1  index ← []
2  clusters ← []
3  foreach document d in the stream do
4  |  S(d) ← ∅                    // set of documents that share a term with d
5  |  foreach term t in d do
6  |  |  foreach document d' in index[t] do
7  |  |  |  S(d) ← S(d) ∪ d'
8  |  |  end
9  |  |  index[t] ← index[t] ∪ d
10 |  end
11 |  c_max ← 0          // maximum cosine between d and documents in S(d)
12 |  n_d ← nil                    // document with maximum cosine to d
13 |  foreach document d' in S(d) do
14 |  |  c := cosine(d, d')
15 |  |  if c > c_max then
16 |  |  |  c_max ← c
17 |  |  |  n_d ← d'
18 |  |  end
19 |  end
20 |  if c_max ≥ m then
21 |  |  add d to clusters[n_d]
22 |  else
23 |  |  clusters[d] ← new cluster(d)          // Report d as a new event
24 |  end
25 end
```

At a high level, almost all of the approaches proposed by the TDT project follow the same nearest-neighbour clustering approach shown in algorithm 1. As documents appear in the stream, they are compared to every document that has been seen before (or, if an inverted index is used, than all documents they share a term with), and the similarity between the new document and every other document in the Corpus is calculated. If one or more similar documents is found (based on some pre-defined

threshold), add the new document to cluster containing the closest match, signaling that they discuss the same event. If no similar documents are found, then a new event has been detected, and a new cluster is formed [Allan et al. 2000a].

Unfortunately, the approach shown in algorithm 1 does not scale to extremely high volume streams. The approach takes $O(|D_t|)$ time to compute in the worst case, and although the use of an inverted index helps to reduce the average case, it continues to take increasingly longer to process new documents as new data is seen. This has obvious drawbacks when the volume of data is increased from a few thousand documents, to many hundreds of millions of documents per day, which quickly makes this approach computationally infeasible.

This basic model persisted throughout the TDT project, and variations of this model (with efficiency optimizations) are commonly found in event detection approaches for social media [Becker et al. 2011b; Petrović et al. 2010b; Aggarwal and Subbian 2012; Ozdikis et al. 2012].

2.2.2 Performance of TDT Systems

The TDT project produced some reasonably effective systems [Allan et al. 2000b; Yang et al. 1998; Allan et al. 2005], however performance was still far below that required for systems to be considered adequate for complete automation [Allan et al. 2000a]. When the TDT project ended in 2004, research in event detection slowed. It is believed that a plateau had been reached, and that without a radically different approach, the performance of these systems was unlikely to ever improve significantly [Allan et al. 2000a]. This belief has remained true in the context of newswire documents, and many state-of-the-art systems are now over a decade old [Allan et al. 2000b; Yang et al. 1998; Allan et al. 2005]. The TDT project was a cornerstone in the development of event detection approaches and is responsible for much of the foundation on which many state-of-the-art event detection approaches for Twitter are based.

2.3 Event Detection on Social Media

In recent years, interest in event detection has been rekindled as social media data has become available for research. However social media posts introduce their own unique challenges which require novel approaches to overcome. The massive volume of data means that traditional event detection techniques are too slow for real-time applications. One of the requirements of an effective event detection system is the

ability to monitor events in real-time. In addition, with high volume data streams, as is the case with Twitter, efficiency is crucial to an effective system.

Analysis of social media has received a lot of attention from the research community in recent years. However, much of this work has focused on blog and email streams [Zhao et al. 2007; Jurgens and Stevens 2009; Becker et al. 2010; Nguyen et al. 2012], using datasets such as the Enron email dataset Klimt and Yang [2004] or the BlogLines dataset [Sia et al. 2008]. More recently, focus has moved towards Twitter due to its popularity with individuals and organizations as a method of broadcast communication.

Hu et al. [2012] demonstrated the effectiveness of Twitter as a medium for breaking news by examining how the news of Osama bin Laden's death broke on Twitter. They found that Twitter had broken the news, and as a result, millions already knew of his death before the official announcement.

Kwak et al. [2010] analyzed the top trending topics to show that the majority of topics (over 85%) are related to headline news or persistent news. They also found that once a tweet is retweeted, it can be expected to reach an average of 1,000 users.

Osborne et al. [2012] measured the delay between a new event appearing on Twitter, and the time taken for the same event to be updated on Wikipedia. Their results show that Twitter appears to be around 2 hours ahead of Wikipedia. They suggest that Wikipedia could be used as a filter, decreasing the number of spurious events, however at the cost of greatly increased latency. They demonstrate its effectiveness as a filter for their First Story Detection approach [Petrović et al. 2010b], significantly decreasing the number of spurious events.

2.4 Event Detection on Twitter

A number of excellent survey papers covering Event Detection on Twitter have been published in recent years, including Hasan et al. [2017], Goswami and Kumar [2016], and Atefeh and Khreich [2015]. Rather than replicate their work here, we invite interested readers to refer to these papers for a full survey all event detection approaches for Twitter. Instead, we survey only the most novel and relevant work in this section.

Sankaranarayanan et al. [2009] proposed one of the first systems which aimed to detect breaking news and events from tweets. In conjunction with the Twitter gardenhose, they utilize a set of 2,000 handpicked *seeds* – twitter accounts who primarily

post news-related tweets – that they used as a trusted source of news. An initial layer of filtering is performed on all incoming tweets, excluding those from the seeds, using naive Bayes classifier trained on a corpus of news and "junk" tweets. Next they use an online clustering approach that maintains a list of active clusters, along with an associated TF-IDF [Salton and Buckley 1988] feature-vector of the terms used by tweets in the cluster. An incoming tweet is added to a cluster if its similarity with the cluster's feature-vector is above a set threshold, measured using a time-decayed cosine similarity function. An inverted index is maintained so that only active clusters which contain at least some of the terms from the incoming tweet are used for comparison, and clusters with a time centroid greater than 3 days old are marked as inactive and removed from the index. To further reduce the number of noisy clusters, Sankaranarayanan et al. [2009] impose an interesting restriction, in that for a cluster to remain active after k tweets have been added, one of the tweets must come from a seed account. Despite a number of efficiency optimizations, their approach is unable to scale, or produce results in real-time. Unfortunately, no attempt is made to evaluate the effectiveness of their system other than a small number of empirical observations.

Becker et al. [2011b] use the clustering approach proposed by Yang et al. [1998] as part of the TDT project, and a filtering layer similar to Sankaranarayanan et al. [2009]. However, filtering is performed after clustering has taken place, and they attempt to identify likely event clusters, rather than event tweets. They use a number of features, such as top terms, and number of retweets, to classify clusters as either event or non-event. Although evaluation of their approach seems to show that it is very effective, it is still based on a model designed for much lower volume streams, and is simply too slow to scale past very small corpora.

2.4.1 Locality Sensitive Hashing (LSH) using Random Hyperplanes

Earlier we discussed how the basic TDT approach as described in Section 2.2.1 could not scale to extremely high volume streams (even with the aid of an inverted index), making it infeasible to use for event detection on Twitter. Petrović et al. [2010b] proposed a solution to this using a technique called Locality Sensitive Hashing (LSH). LSH uses a special hash functions that produce the same hash for documents that are similar, but not necessarily completely identical, allowing documents that are close in vector space to be placed into the same bucket. Using a set of random hyperplanes and a family of hash functions proposed by Charikar [2002], the buckets are defined by the subspaces between hyperplanes, such that the probability of sim-

ilar documents being placed into the same bucket of a hash table is proportional to their similarity.

By using multiple hash tables, each with independently chosen random hyperplanes, the probability of the true nearest neighbour colliding in at least one table can be increased to any desired probability (at the cost of additional computations). The number of hash tables L needed to give the desired probability of *missing* a nearest neighbour δ can be computed as:

$$L = log_{1-P_{coll}^{k}}\delta$$

where k is the number the number of hyperplanes, $P_{coll} = 1 - \frac{\theta(x,y)}{\pi}$ where $\theta(x, y)$ is the expected angle between similar documents x and y. Petrivic et al. use values of $k = 13$, $\theta(x, y) = 0.2$ and $\delta = 0.025$ to compute L, values which we also use when replicating their work in Chapter 3.

Variance Reduction

Although the use of LSH improves computational efficiency, it proves to be considerably less effective than the standard TDT approach. This is because LSH is most effective when the true nearest neighbour is very close (i.e. has a high similarity) to the query document. If the nearest neighbour is far from the query document, then LSH often fails to find it. To overcome this, Petrović et al. [2010b] use a 'variance reduction' strategy when a nearest neighbour is not found using LSH. In these instances, the approach falls back the traditional TDT model, and uses an inverted index to retrieve a list of documents to search for a nearest neighbour. However, unlike the basic TDT model, a limit is placed on the number of documents searched, and only the most recent 2000 documents returned from the inverted index are compared. This helps to improve the effectiveness, and brings it inline with that of the basic model, whilst still being considerably more efficient.

Additional Efficiency Enhancements

Since the number of buckets is limited, in a streaming scenario where the number of documents in unbounded, the number of documents in each bucket will continue to grow as new documents are processed. This would require an unbounded amount of space and the number of comparisons made would also grow for each new document, eventually making even the LSH approach infeasible. To overcome this, Petrović et al. [2010b] limit the number of documents in a bucket to a fixed number based on the expected number of collisions, which can be computed as $n/2^k$ where n is the

total number of documents and k is the number of hyperplanes used. When a bucket is full, the oldest document in the bucket is removed from that bucket (and only that bucket) to make room for new documents. Due to the use of multiple independent hash tables, the removal of the document from a bucket does not prevent it from being retrieved from other hash tables, although eventually all documents are removed from all hash tables.

To keep the number of comparisons per document fixed, similarity comparisons are performed for at most $3L$ documents (where L is the number of hash tables). The documents are selected by counting the number of times a document collides across the L hash tables, and taking the top $3L$ documents with the most collisions.

Cluster Ranking

Although the use of LSH improves computational efficiency and allows the application of a TDT approaches on Twitter-scale data, it does not address the issue of noise and spam, which is found in excess on Twitter but is not present in the TDT datasets.

Petrović et al. [2010b] investigated a number of ranking approaches for ranking clusters produced by their system. They found that ranking clusters by the number unique users was more effective than the raw number of tweets. They also found that the amount of information contained in the thread, measured using Shannon entropy [Shannon 2001], was a good indicator of quality. Shannon entropy, H, is measured as:

$$H = -\sum_i \frac{n_i}{N} \log \frac{n_i}{N}$$

where n_i is the number of occurrences of term i in a cluster, and $N = \sum_i n_i$ is the total number of all term occurrences in the cluster. Clusters with a low entropy, defined as $H < 3.5$ by Petrović et al., are moved to the bottom of the ranking – effectively marking them as noise.

Note that this ranking approach requires either a fixed period to have elapsed or a set number of tweets to be processed before the ranking can be made. For example, the ranking may be updated once per hour, or once every 100,000 tweets. This takes away from the usefulness of the approach in a real-time scenario, making it more batch-based than real-time.

2.4.2 Efficient Clustering Using a Fixed Number of Clusters

If documents are restricted to a single cluster, an easy efficiency gain can be found by comparing new documents to existing clusters (rather than every other document). Since the number of clusters will always be smaller than the number of documents, fewer comparisons are needed to find the nearest neighbour cluster. This approach was used by Aggarwal and Subbian [2012], with the additional constraint that the number of clusters (usually several hundred) is kept constant, ensuring linear complexity (i.e., the time taken to process each new document is a constant). An overview of their clustering approach is given in Algorithm 2.

ALGORITHM 2: A clustering approach as given by Aggarwal and Subbian [2012] with a fixed number of clusters

Input: Number of clusters k

clusters $\leftarrow [k]$
$i, \mu, \sigma, M_0, M_1, M_2 \leftarrow 0$
foreach *document d in the stream* **do**
    $s_{max} \leftarrow 0$ `// maximum similarity between` d `and all clusters`
    $n_c \leftarrow nil$ `// cluster with maximum similarity to` d
    foreach *cluster c in clusters* **do**
        $s \leftarrow$ similarity(d, c)
        if $s > s_{max}$ **then**
            $s_{max} \leftarrow s$
            $n_c \leftarrow c$
        end
    end
    if $s_{max} < \mu - 3 \cdot \sigma$ **then**
        replace most stale cluster with new cluster containing d
    else
        add d to clusters[n_c]
    end
    update M_0, M_1, M_2 additive
    $\mu \leftarrow M_1 / M_0$
    $\sigma \leftarrow \sqrt{M_2/M_0 - \mu^2}$
end

Identifying Event Clusters

Aggarwal and Subbian [2012] define two ways that a new event can occur using their clustering approach:

- *novel* events, which are caused by the creation of a new cluster containing only a single document (thus, also requiring the removal of a "stale" cluster)
- *evolution* events, where a cluster experiences a rapid increase in the relative volume of documents it receives due to the emergence of a new topic

New clusters (*novel* events) are created when a nearest neighbour cluster cannot be found among the existing clusters. Unlike most event detection and TDT systems, Aggarwal and Subbian [2012] use an adaptive similarity threshold based on the Three Sigma rule [Pukelsheim 1994], which states that the expected similarity value should be within 3 standard deviations of the mean similarity across all previously seem documents. For use as a minimum threshold for similarity, only the lower bound is used:

$$threshold = \mu - 3 \cdot \sigma$$

where μ is the mean and σ is the standard deviation of all similarity measures that have been seen before.

To compute the μ and σ values efficiently, Aggarwal and Subbian [2012] use the first 3 moments of the set of highest similarity scores S (i.e., the set of similarity score of each document to its nearest neighbour cluster):

$$M_0 = \sum_{i=0}^{|S|} S_i^0 \qquad M_1 = \sum_{i=0}^{|S|} S_i^1 \qquad M_2 = \sum_{i=0}^{|S|} S_i^2$$

These values can be updated as each new document is clustered, and used to calculate the μ and σ as show in Algorithm 2.

If a document has a highest similarity score below $\mu - 3 \cdot \sigma$ then a new cluster is created and the document is added to it. Since this approach requires a constant number of clusters for linear time complexity, the creation of a new cluster requires the removal of an existing cluster. Aggarwal and Subbian [2012] opt for the simplest approach in this case, and remove the cluster which has not had a document added for the longest period of time.

An *evolution* event is said to have occurred when the fraction of documents added to a cluster during time t_i is greater than the fraction added during period t_{i-1} by a factor of α. For a cluster c, during time periods t_i and t_{i-1}, an evolution event occurs if:

$$\frac{F(c, t_i)}{F(c, t_{i-1})} \geq \alpha$$

where F gives the fraction of all documents during time t that were added to cluster c. Evolution events are necessary to capture real-world events that are similar enough to an existing cluster that they do not cause a new cluster to be created. The change in relative volume of documents represents a change in behavior which can be seen as an indicator of a new event.

2.4.3 Supervised Approaches to Event Detection

Sakaki et al. [2010] were concerned with the detection of specific types of event, in particular, earthquakes and typhoons, with the aim of issuing alerts to those in the path of these disasters. Their approach, although simple, is very effective. They specify a set of predefined keywords which are associated with the type of events they are trying to detect (e.g. earthquake, shake, cyclone, typhoon, etc.,). They monitor an incoming stream of tweets for these keywords, and for each tweet found, they classify it using a Support Vector Machine (SVM) as either event-related or not. If they find enough event-related tweets in a short period of time, then their systems decides that the event is real, and issues alerts to those who could be affected. Despite the effectiveness of their approach, there are obvious drawbacks. Firstly, it can only detect specific types of event, and secondly, it requires training data for each of the types of event we want to detect.

2.4.4 Natural Language Approaches

Choudhury and Breslin [2011] examined how linguistic features and background knowledge could be used to detect specific types of sports events with high precision. However their approach requires significant domain knowledge of the sporting events and large amounts of manual labeling to prepare a classifier for each specific type of event, making it difficult and time-consuming to generalize.

Ritter et al. [2012] used named entities, “event phrases” and temporal expressions to extract a calendar of significant events from Twitter. However, their approach requires that tweets contain temporal resolution phrases, such as “tomorrow” or “Wednesday” for their approach to resolve between an entity and a specific time. This means that smaller events, which do no generate much discussion before they happen, are unlikely to be detected. Additionally unexpected events, which are often the events which are of most interest, are unlikely to be detected as they will be discussed as they happen, and without any temporal resolution phrases.

Most similar to our work is that of Li et al. [2017], who use a partitioning technique to efficiently cluster documents by splitting the tweet term space into groups they call ‘semantic categories’, such as named entities, mentions, hashtags, nouns and verbs. To ensure that only novel events are reported, they compare each new event to old events, filtering out any events that have already been reported. They merge events by comparing the top 20% of entity terms, based on term frequency, which reduces event fragmentation. They evaluate their approach using the Events 2012 corpus we create

in chapter 3, finding that their approach outperforms the LSH approach when measured using NMI and B-Cubed cluster performance metrics. Although this approach outperforms their baseline approach, the LSH approach of Petrović et al. [2010b] using their chosen evaluation metrics, they do not report precision or recall values, making it difficult to compare directly.

2.4.5 Burst and Trend Analysis

2.5 Differences between Event Detection and other IR Tasks

Despite being viewed as an IR task, Event Detection has a number of characteristics that differ from more common IR tasks. The most obvious of these is the lack of a user-provided query. Rather than searching for documents relevant to a specific query, event detection systems first must detect new events and track which documents belong to the same topic. This means that the notion of relevance is much more ambiguous, and creates a number of issues with the evaluation of event detection approaches.

2.6 Test Collections

Ir order to fairly and accurately evaluate Information Retrieval systems of any kind, a standard test collection and set of relevance judgements are required. The majority of modern test collections are created using a pooling approach, where a set of topics or queries are selected and used to retrieve documents from a number of different systems. The union of the top n documents per topic, taken from each system systems, is then used create a pool of documents for each topic. Each document in the pool is then judged to be relevant or non-relevant to the topic, and documents which are not in the pool are assumed to be non-relevant.

This pooling approach is commonly used to generate relevance judgements for collections provided to researchers participating in conferences such as the Text REtrieval Conference (TREC), which is sponsored by the National Institute of Stands and Technology (NIST) in the United States. Researchers are first given the test collections with a set of pre-selected topics, and are asked to submit the results from their retrieval systems to experts who then perform the pooling and annotate the documents. Given the scale of modern test collections, this can be a slow and expensive undertaking, which restricts the number of “tracks” (a set of tasks focused on some facet of a retrieval

problem) that can be run each year, and thus limits how many test collections and relevance judgments can be created this way.

More recently, crowdsourcing has become a popular method of cheaply and quickly generating relevance judgements since it does not rely on the timing of particular conferences or and does not require experts to gather relevance judgements. Alonso et al. [2008] describe how crowdsourcing can be used to gather relevance judgements and discuss their crowdsourcing methodology called 'TERC', which they find provides fast turnarounds whilst being low cost and producing high quality results. However, they also describe some of the issues and limitations which need to be addressed, such as the need for strict quality control and the 'artificiality' of the task. Whilst they propose some solutions to these issues, they recognize that many of the solutions are domain specific, and not applicable in all situations.

Issues remain with the pooling method itself. Buckley et al. [2007] highlighted a number of issues associated with pooling when dealing with large collections, and showed that standard pooling techniques can introduce bias when dealing with large collections. They make a number of suggestions, such as engineering the topics or forming pools differently, that could reduce bias and allow for the creation of large unbiased test collections using pooling. However, the use of pooling to create a test collection for event detection poses additional challenges which require more substantial changes to the TREC-style pooling methodology than those proposed by Buckley et al. [2007].

2.6.1 The Pooling Approach and Event Detection

As discussed in section 2.5, there are no pre-defined topics or queries for event detection systems to use. Rather, systems aim to automatically *discover* a sub-set all all topics being discussed in the collection that meet the criteria for being an *event* (as we will discuss in Chapter 3, defining these criteria is not straightforward). A perfect system would detect all documents for each event being discussed, however in practice, different event detection approaches discover different events. This makes it impossible to pool the results from event detection approaches using a TREC-style pooling methodology since it is not obvious what the various topics are or how they should be merged to create the pools for judgment.

Even if it was possible to identify and pool tweets for each of the topics, event detection systems generally do not produce a rank for each tweet, instead relying on their temporal order. Temporal order is a natural and useful way of ordering tweets that

discuss real-world events, as events are often discussed in real-time. Since the rank does not infer any sort of quality or usefulness score from the system, there is no way of identifying the top n documents from each topic from each system, making it difficult to pool them effectively.

2.6.2 Twitter Corpora

Table 2.1: A comparison of the different Twitter corpora available prior to this collection. Values marked with * are estimates as exact numbers are not given. A question mark (?) indicates that the number of events is not clear.

Collection	Period	Tweets	Event based	Events	Unbiased
McCreadie et al. [2012]	14 days	16M	No	-	-
Becker et al. [2012]	28 days*	2.6M	Yes	?	No
Petrović et al. [2012]	75 days*	50M	Yes	27	No
This Work	28 days	120M	Yes	506	Yes

In this section, we examine publicly available Twitter corpora, paying particular attention to their suitability for the evaluation of event-based systems and analysis. We also discuss some of the challenges associated with the creation of a Twitter-sized corpus.

TREC ran the Microblog Track in 2011 with an ad-hoc retrieval task, and again in 2012 with ad-hoc retrieval and filtering tasks. The Tweets2011 corpus McCreadie et al. [2012] was used both years, with new relevance judgements being generated each year. The collection was the first publicly available large-scale Twitter corpus, containing 16 million tweets which cover a period of 2 weeks. However, the corpus contains tweets in all languages and only around 4 million tweets remain after non-English tweets are removed. Furthermore, the corpus is designed specifically for ad-hoc retrieval, and as such, the topics and relevance judgements are unsuitable for event-based analysis. The TREC Microblog Track also ran in 2013, however the track moved to an experimental track-as-a-service approach, where the corpus was hosted by TREC and participants had to be queried using an API. Once again, the set of topics used were designed specifically for ad-hoc retrieval, making them unsuitable for event-based analysis.

Becker et al. [2012] produced what we believe was the first Twitter collection of events, however it only contains tweets posted by users in New York. This clearly introduces geographical bias and restricts the type of events available. The collection itself is also relatively tiny, containing only 2.6 million tweets.

Petrović et al. [2012] created a corpus aimed at First Story Detection, and while their collection contains a relatively high 50 million tweets from the beginning of July until mid-September 2011, they identified only 27 events. This means that large scale comparisons are difficult as there is not a large sample of events (less than 1 event per day) and failure to detect only a small number of these could result in unsubstantiated and misleading results.

Although these collections have been made available, albeit in a limited fashion, none appear suitable for the analysis of events and comparison of event detection approaches. The collection produced by Becker et al. is simply too small to be of practical use for event detection, while the collection created by Petrović et al. covers too few events for a fair comparison to be made. One reason for the lack of comparative corpora may be the difficulty and expense of creating one. A reasonable sized Twitter corpus will contain tens of millions of documents – performing a manual search on corpus of that magnitude is simply impossible. To overcome this, Petrović et al. used a procedure similar to NIST, where expert annotators (primarily the authors of the paper) read the description of an event and used keywords to search for relevant documents. However, this approach means that events (i) must be carefully identified in advance (potentially introducing bias) (ii) annotation requires expensive and slow experts, and (iii) it does not scale well past a certain size (Petrović et al. were only able to create judgements for 27 events). This demonstrates the need for a methodology of creating unbiased large-scale and real-world corpora which can be used for the comparison of event detection techniques.

Issues with both TREC-style pooling and crowdsourcing highlight the need for a new methodology which can be used for the creation of large-scale test collections for event detection on microblogs. A new type of pooling is needed in order to cope with the lack of predefined topics and document ranking, whilst new crowdsourcing techniques are needed which maintain the fast turnarounds and low-cost associated with crowdsourcing, whilst ensuring high-quality results.

CHAPTER 3

Building a Twitter Corpus for Evaluating Event Detection

Event Detection is one of the most commonly studied tasks on Twitter [Aggarwal and Subbian 2012; Becker et al. 2012; Zhao et al. 2007; Weng and Lee 2011; Becker et al. 2011a; Petrović et al. 2010b; Li et al. 2012b]. Twitter provides a real-time stream of updates, opinions, and first-hand reports of what is happening, all of which are difficult or impossible for the traditional newswire to match. This makes it very desirable to develop systems which are able to detect and track events from Twitter streams. However, creating a system which is able to detect events in real-time is hard, not just because of the difficulty of the task [Allan et al. 2000a], but also due to the speed and efficiency needed to process such high-volume data streams. There is also disagreement on the definition of 'event', which makes comparisons of different event detection approaches very difficult – one system may consider something to be a single event, while others may break it down into multiple events, or may not regard it as an event at all. For example, a football game could be considered a single event or broken into individual plays – either is reasonable depending on the type of event you are trying to detect. Additionally, there is no standard corpus for the evaluation of different event detection approaches, and most works require the creation of a bespoke corpus, which is often not made available. This results in wasted time and resources, motivating the need for a new methodology and a corpus which can be used for the evaluation and comparison of event detection approaches.

Despite the interest in Twitter, there are only a small number of corpora available, none of which are suitable for the large-scale evaluation of event detection approaches due to their small size or number of topics. This is partially due to the massive scale of Twitter, which makes the creation of corpora difficult, time-consuming and expensive, but also due to disagreement on the definition of event, which can often make comparisons difficult or impossible. Furthermore, the Twitter Terms of Service restrict the distribution of tweets, and do not allow the content of tweets to be distributed as part of a corpus. As a result, there are very few publicly available Twitter cor-

pora, and in some cases, corpora from other medium are used in place of a Twitter corpus [Aggarwal and Subbian 2012; Petrović et al. 2010b, 2012]. However, it is not clear that effectiveness of event detection approaches on a non-Twitter corpus is comparable to effectiveness on a Twitter corpus, with some evidence suggesting that this is not the case [Petrović et al. 2012]. This scenario leads to a situation in which event detection techniques are not properly benchmarked.

To solve this, we propose a new, Twitter centric, definition of event which we believe better fits how users perceive events and how events are discussed on Twitter. Then, using this new definition, we study the problem of creating a set of relevance judgements for the evaluation of event detection approaches. To do so, we use two existing event detection approaches and the Wikipedia Current Events Portal to generate a set of candidate events and associated tweets. We then use crowdsourcing to evaluate if a candidate event fits our definition of event, and create relevance judgements for each of the events. We then merge events from each of the sources which discuss the same real-world event, and extending our original work [McMinn et al. 2013], examine the quality of the clustering. Finally, we make the corpus and judgements available for research and further development.

This chapter has a number of novel contributions:

- We create the first test collection of this scale for event detection on Twitter
- We examine existing definitions of "event", and propose more robust definition that deals with the nuances of events on Twitter
- We propose a novel methodology for the creation of relevance judgements using two state-of-the-art event detection approaches, Wikipedia, and Mechanical Turk
- We study the quality and characteristics of the corpus and the relevance judgements

3.1 Definition of Event

Despite the significant interest in events, there is little consensus on the exact definition of *event*. This leads to issues when evaluating and comparing event-based systems.

The *Topic Detection and Tracking* (TDT) project defines an *event* as 'something that happens at some specific time and place, and the unavoidable consequences' [Allan 2002a]. Specific elections, accidents, crimes and natural disasters are examples of

events under the TDT definition. They also define an *activity* as a connected set of actions that have a common focus or purpose. Specific campaigns, investigations, and disaster relief efforts are examples of activities. Furthermore, a *topic* is defined as a seminal event or activity, along with all directly related events and activities. The TDT project dealt with newswire documents, implying a level of significance to the topics being discussed – it was reasonable to assume that the vast majority of topics in the TDT datasets were significant events. However this is not the case in Twitter, where a very large portion of documents discuss insignificant personal matters, such as the song a user is listening to. The TDT project was concerned with the detection of *topics*, the US Presidential Elections is considered a single topic, and stories about the candidates' campaigns, debates, election results, and reactions to the election are all part of the same topic. There is no distinction made between each of the events within a topic, even though they could occur several days apart. We believe that this does not make sense in the context of Twitter as discussion moves very quickly and is fixated on the present, unlike newswire documents, which often summarize several days worth of events into a single article.

Aggarwal and Subbian [2012] define a *news event* as being any event (something happening at a specific time and place) of interest to the media. They also consider any such event (e.g. a speech or rally) as being a single episode in a larger story arc (i.e. a presidential campaign). They use the term *episode* to mean any such event, and *saga* to refer to the collection of events related within a broader context.

Becker et al. [2011a] go as far as to define an event in a much more formal, but still subjective manner. They define an event as a real-world occurrence e with (1) an associated time period T_e and (2) a time-ordered stream of Twitter messages, of substantial volume, discussing the occurrence and published during time T_e. Other definitions, such as that used by Weng and Lee [2011], define an event simply as a burst in the usage of a related group of terms.

Clearly there is a consensus that events are temporal, as time is a reoccurring theme within all definitions. However, the consensus appears to end there. Whilst Aggarwal and Subbian [2012] and the TDT definition [Allan 2002a] show a parallel in their hierarchical organization of events (events and topics, news events and sagas), this is less common in other definitions where a distinction between events and topics is not made. This makes comparisons very difficult; one definition may break an election into many events, while another could consider the election as a single event, or not as an event at all.

To solve these issues, we take the most basic definition of event (something that happens at a particular time and place), and introduce the requirement that an event should be *significant*. By requiring that an event is significant, we are able to filter out every-day personal and trivial events which are extremely common on Twitter (such as getting a phone call or going to the gym).

Event: An event is a *significant* thing that happens at some specific time and place.

As discussed previously, it was reasonable to assume that all events in the TDT datasets were significant events due to the use of newswire documents, something which is not true in the case of Twitter. Given this, we attempt to model our definition of significance so that an event under our definition would be of a similar level of significance to those found in the TDT datasets, despite the disparity between the 2 sources.

Significant: Something is significant if it may be discussed in the media. For example, you may read a news article or watch a news report about it.

It is important to note that something does not necessarily have to be discussed by the media in order for it to be an event, we simply use this as an indication of the level of significance required for something to be considered an event. Whilst this is still somewhat subjective, we believe that it is impossible to further restrict significance whilst keeping our definition of event generalizable. Given this definition of event, our aim is to create a collection of significant events which have been discussed on Twitter, and set of relevance judgements for tweets which discuss the events.

By using the term *significant* to qualify a topic as an event we allow for the definition of event to be kept constant event for different use-cases. While we align our definition of *significant* such that it brings out definition of event in line with that of the TDT project, we do not exclude the possibility that *significant* could be altered to suit the particular use-case. For example, a system designed to assist the emergency services may only consider an event to be significant if it requires an emergency response, whereas a system designed for financial traders may define significant as something that might affect the price of a security.

3.2 Building the Corpus

We collected over 1 billion tweets from the Twitter Streaming API[1] for 28 days, starting on the 10th of October and ending on the 7th of November 2012. This period was chosen specifically because it covers a number of interesting and significant events, including Hurricane Sandy and the U.S. Presidential Elections. Language filtering was performed using a language detection library for Java[2] to remove non-English tweets. Further filtering was performed to remove common types of Twitter spam (i.e. tweets which contain more than 3 hashtags, more than 3 mentions, or more than 2 URLs, as these have been shown to be strongly associated with spam [Benevenuto et al. 2010]). After spam and language filtering, we were left with a corpus of 120 million tweets.

Of the 120 million tweets in the corpus, around 30% (40 million) are *retweets*. A retweet is a copy of another user's tweet which was broadcast by a second user to their followers, often prefixed with 'RT'. In the context of Twitter, retweets are a very useful method of spreading information. However, retweets are commonly associated with the spread of spam [Boyd et al. 2010], and because they are an unmodified copy of another user's tweet, they do not generally add any new information. Given this, and in order to reduce the complexity of creating relevance judgements, we chose not to include retweets in the relevance judgements.

The remainder of this section details the approach used to generate a list of events and relevance judgements for the corpus. We begin by describing the approach used to generate a set of candidate events and associated tweets. We then describe the crowdsourced evaluation and discuss a number of spam prevention and detection methods.

3.2.1 Candidate Event Generation

Rather than create our own list of events, we chose to use a number of existing event detection approaches and the Wikipedia Current Events Portal[3] to create a pool of events. In the remainder of this chapter, we refer to the event detection approaches as *detection approaches* and the use of the Wikipedia Current Events Portal as the *curated approach*.

Event detection in microblogs is very similar to the clustering task [Allan 2002b] (more

[1] https://dev.twitter.com/docs/streaming-apis
[2] https://code.google.com/p/language-detection/
[3] http://en.wikipedia.org/wiki/Portal:Current_events

commonly referred to as *detection*) in TDT. In both, a system is presented with a continuous stream of time ordered documents and must place each of them into the most appropriate event-based cluster. The main difference between the two tasks is the type and volume of documents in the stream – however in practice this makes a great deal of difference and event detection on microblog streams is considerably more challenging for a number of reasons. Firstly, the volume of documents is several orders of magnitude greater in microblogs, which means that event detection systems must be extremely efficient and preferably able to run in real-time. Secondly, the majority of microblog posts discuss mundane, every day occurrences which are not noteworthy or of interest. These documents must be filtered out so that only interesting real-world events are detected, an issue which was not present for participants of the TDT evaluations. Furthermore, microblog posts tend to be very noisy, of very limited length[4] and frequently contain spelling or grammar errors. These differences mean that approaches developed for the TDT program tend to be too slow for real-time application, and extremely vulnerable to the noise found in microblog streams.

Given these issues, we choose to use two state-of-the-art detection approaches (at the time of this work was carried out), which were designed specifically for Twitter, namely the Locality Sensitive Hashing approach proposed by Petrović et al. [2010b] and the Cluster Summarization approach proposed by Aggarwal and Subbian [2012]. These were chosen based upon a number of desirable characteristics. Firstly, both approaches produce clusters of documents, rather than clusters of terms. Many event detection approaches for Twitter attempt to reduce the problem by clustering terms rather than documents. However, this is considerably less useful in our case as they are much more difficult to evaluate and would require an additional step in order to retrieve tweets. Secondly, both approaches are efficient and we were confident that they would finish in a reasonable time frame (i.e. days rather than weeks or months). Whilst it would have been desirable to implement additional detection approaches, the time taken to implement, run and evaluate each approach is prohibitive. Furthermore, since most detection approaches use similar methods to cluster documents and detect events, it is doubtful that a small increase in the number of detection approaches used would significantly increase the coverage of events.

In addition, Wikipedia maintains a Current Events Portal, which offers a curated set of events from the around the world. The use of Wikipedia offers a number of advantages over the use of additional detection approaches. Firstly, each of the events listed

[4]Tweets were restricted to 140 characters at the time the collection was created. Twitter have since increased the limit to 280 characters.

on the current events portal is substantiated by a link to a news article from a reputable news source. This allows a high level of confidence that the events are accurate and significant under our definition. Secondly, much of the work has already been done for us by unpaid editors and is of a high quality, ensured by Wikipedia's editorial guidelines. This means that we do not have to pay workers to evaluate non-event clusters, reducing the cost and time taken to produce judgements. In addition, Wikipedia provides complimentary results to the detection approaches thanks to its wide coverage and diversity.

The remainder of this section details both detection approaches and the curated approach, and describes how they were used to generate a set of candidate events.

Locality Sensitive Hashing (LSH)

Petrović et al. [2010b] solve the issue of efficiency using a technique called Locality Sensitive Hashing (LSH), which places similar documents into the same bucket of a hash table.

Documents are hashed using a certain type of hash function:

$$S(x) = \{y : h_{ij}(y) = h_{ij}(x), \exists i \in [1..L], \forall j \in [1..k]\}$$

where hash functions h_{ij} are defined as:

$$h_{ij}(x) = sgn(\mathbf{u}_{ij}^T x)$$

with the random hyperplanes $\mathbf{u}_{ij}$ being drawn independently from a Gaussian distribution for each i and j.

Using multiple hash tables it is possible to reduce the size of the candidate set to a fixed size which contains the nearest neighbour with a high probability. Clustering can then be performed in $O(1)$ time. In cases where no neighbour is found within a certain distance, a variance reduction technique is used which has been shown to vastly improve clustering effectiveness [Petrović et al. 2010b].

Every 100,000 tweets, the clusters are ranked based upon the number of new unique users, which the authors found outperformed other measures, such as number of tweets. In order to filter out noise and spam, Shannon entropy is used to measure the amount of information in the cluster, and clusters with small entropies (<2.5) are moved to the back of the list of possible events.

We ran the algorithm over our corpus using parameters very similar to those used by Petrović et al. [2010b]. More precisely, 13 bits per key, a maximum distance of 0.45 and 70 hash tables. However, we chose to measure the fastest growing clusters on a hourly basis, rather than every 100,000 tweets as used in the original paper. We made this decision due the fact that 100,000 tweet covers only a short period of time in our collection (around 30 minutes) due to its high density. Since the approach keeps several hours worth of tweets in the hash table at all times, this would have generated twice as many candidate events without necessarily increasing the number of actual events, making it prohibitively more expensive to generate judgements. Simply taking the list of events every hour would have yielded a prohibitively high number of clusters. However by removing clusters outside the optimal entropy range (3.5 - 4.25) [Petrović et al. 2010b], and by removing clusters with less than 30 tweets (which ensures the cluster contains enough tweets for effective evaluation), the list is reduced to a manageable size of 1340 candidate events.

Cluster Summarization (CS)

Aggarwal and Subbian [2012] use a fixed number of clusters and cluster summaries to reduce the number of comparisons required for document clustering. Each cluster summary contains (i) a node-summary, which is a set of users and their frequencies and (ii) a content-summary, which is a set of words and their TF-IDF weighted frequencies. By combining these two summaries, they suggest a novel similarity score which exploits the underlying structure of the social network, and improves upon content-only measures. A sketch-based technique is used to maintain node statistics and is used to calculate structural similarity at a significantly reduced cost.

The similarity score between document D and cluster C is given by:

$$Sim(D, C) = \lambda \cdot SimS(D, C) + (1 - \lambda) \cdot SimC(D, C)$$

Where $SimS$ and $SimC$ are structure and content similarity methods respectively, and λ is a balancing parameter in the range (0,1).

Each incoming document is assigned to its closest cluster unless its similarity score is significantly lower than that of other documents. A similarity score is considered significantly lower if it falls below $\mu - 3 \cdot \sigma$, where μ is the mean of all previous similarity scores, and σ is the standard deviation [Pukelsheim 1994].

We ran the CS algorithm [Aggarwal and Subbian 2012] with an input size of 1200 clusters,

slightly more than used by Aggarwal and Subbian [2012]. This gave a reasonable runtime of approximately 4 days for the entire collection on the hardware available to us, whilst still maintaining effective clustering performance [Aggarwal and Subbian 2012]. Retweets were not used as input to the algorithm as we found that they tended to cause more spam and noise clusters to form and be identified as events. We used a λ value of 0.0 (i.e. full weight was given to text when calculating similarity) as this greatly decreased the runtime whilst having only a small effect on cluster performance [Aggarwal and Subbian 2012].

Similarly to the LSH approach, we removed clusters with less than 30 tweets, and those with α values smaller than 12 (i.e. slow growth rates) [Aggarwal and Subbian 2012]. Empirically we found that very few clusters with an α value below 12 discussed events, and by removing these clusters we were able to significantly reduce the number of candidate events to a manageable size of 1097.

Wikipedia Current Events Portal (CEP)

The second part of our groundtruth is generated using the Wikipedia Current Events Portal[5], which provides a detailed list of significant events generated by volunteers from around the world. This provides an unbiased list of real-world events and because it does not rely on automated methods, it complements the detection approaches and removes some bias towards easier to detect events.

Each event is represented by a description (at most a few sentences), a category, and a link (or links) to relevant news article. An example event may look similar to:

Date	October 25, 2012
Category	Business & economics
Description	Official [[GDP]] figures indicate the [[2012 Summer Olympic]] helped the [[UK economy]] emerge from recession in the three months from July to September, with growth of 1.0%.
Reference	`http://www.bbc.co.uk/news/business-20078231`

Note that words surrounded by [[and]] are links to other Wikipedia pages.

This approach is very different from the detection approaches, as we already have set of events and want to find relevant tweets. To do this, we indexed the corpus using

[5] http://en.wikipedia.org/wiki/Portal:Current_events

Lucene 4.2[6]. Stop word removal was performed, porter stemming was applied, and prefixes (#, @ characters) were remove from hashtags and mentions.

We then use the description from each of the Wikipedia events as an initial query to retrieve tweets which discussed the event. Query expansion is performed to decrease lexical mismatch, and has been shown to give some of best retrieval performance as part of the TREC Microblog track in 2011 [Amati et al. 2011; Ferguson et al. 2011; Li et al. 2011] and 2012 [Kim et al.; Aboulnaga et al.; Han et al.]. In particular, we expand links to other Wikipedia pages to the full title of that page (e.g. "UK economy" $\rightarrow$ "Economy of the United Kingdom"), and expand/contract acronyms (e.g. "U.K." $\rightarrow$ "United Kingdom", "United States" $\rightarrow$ "U.S."). Furthermore, terms used as links to other pages were given double weighting as they are generally the most important and contextual terms in the description. Divergence from Randomness using Inverse Document Frequency as the basic randomness model was used for retrieval, as experimentation using the TREC11 corpus showed that, of the retrieval models included with Lucene 4.2, it gives the best retrieval performance.

For each of the 468 events on the Wikipedia Current Events Portal listed between the dates covered by the corpus, we retrieved the top 2000 tweets from a window of 72 hours, centred around the day of the event (i.e. for an event on the 16th of October, retrieval was restricted to tweets posted between the 15th and 17th of October inclusively).

3.2.2 Gathering Relevance Judgements

This section describes the methodology used to gather relevance judgements for each of the candidate events and their associated tweets. The objective is to identify which of the candidate events fits our definition of event, and to gather a sample of relevance judgements for each of the events. In addition, we also want to gather high-level descriptions and category information for each of the events, which will be useful for merging the events from the different sources (discussed in section 3.3) and for evaluation purposes.

The following relevance statement was used for all evaluations:

> Anything that discusses the described event is considered relevant, even if the information is now out-of-date or does not necessarily match that

[6] http://lucene.apache.org/

> given in other tweets (e.g. the number of deaths is different). However, care should be taken to mark any untrustworthy or clearly false statements as non-relevant. Tweets which give a user's opinion of an event, and are obviously discussing the event but do necessarily describe what happened, are still considered relevant.

The definition was intentionally very open as we wanted to capture as many tweets about each event as possible. Specifically, as well as objective tweets, we wanted to include subjective tweets (i.e. the opinion of users) as they are one of the main attractions for using social media data when studying events, and are generally not found in newswire documents.

The remainder of this section describes the challenges associated with the gathering of relevance judgements for our collection, including how the number of annotators were selected, how much information was gathered using each HIT (Amazon's name for a single crowdsourced job, which stands for 'Human Intelligence Task'), how categories were defined, and how quality control was performed.

Selecting the Number of Annotators

In order to ensure that noise and spam have minimum impact on the evaluations, multiple annotators need to be used for each evaluation. To help us choose a number of annotators, we ran a pilot using 20 carefully selected candidate events covering a range of categories and with varying degrees of perceived difficult or ambiguity. Several candidate events were selected specifically because they were either difficult to judge or fell between event and non-event (i.e. they were very subjective), while other candidates were selected because they contained a mix of relevant and non-relevant tweets, making them more difficult to judge. Each candidate was annotated by 10 different workers, resulting in 16 of the 20 candidates being identified as an event by a majority ($k = 0.66$).

It is desirable to minimize the number of annotators per candidate event to reduce the cost and time taken to perform the evaluations, however this must be done carefully and without significantly affecting the quality of the results. As a minimum, we need 3 annotators for each event to agree and form a majority. This means that at a minimum, each candidate event needs to be evaluated by 5 annotators. In order to verify that the reduced number of annotators per event gives similar results to the original 10, we randomly selected 5 evaluations for each of the candidates events and compared the results. With 5 annotators per candidate, 17 of the 20 candidates were

identified as events ($k = 0.61$). Whilst this differs slightly from the results obtained using 10 annotators and there is a very slight drop in overall agreement, it requires only half the number of annotators and greatly reduces costs both financially and in terms of time taken. Given the similarity of the results using both 5 and 10 annotators, and the fact that 5 annotators still guarantees a minimum agreement of 3 annotators per event, we chose to use 5 annotators for all remaining evaluations.

Work per Evaluation

One of the biggest issues when designing a crowdsourced evaluation is how much work each HIT should entail. Too much work per HIT and workers will quickly become bored and fatigued, potentially reducing the quality of their annotations. We decided to try and limit the time taken to perform an individual evaluation to 60 seconds, reducing the chance the workers will become bored or fatigued. This meant that we could ask only a limited number of questions in each evaluation, and the number of tweets which we could have annotated in each evaluation was also very limited. In addition to the time taken to answer questions, we also have to consider how much time is required to read the tweets and make a decision about the candidate event (i.e. does the candidate fit our definition of event?), further reducing the amount of questions we could ask in our desired time limit.

Initially, we chose to use 30 tweets, however this caused the time taken to read the tweets and make a judgement about the candidate to be considerably over 1 minute. So solve this, we ran empirical evaluations to measure how many tweets we could read in 20 seconds, leaving 40 seconds to answer questions and annotate the tweets. We found that we were able to carefully read around 13 tweets in a 20 second window, and so used that number for pilots and the evaluation of the detection approaches.

For the Wikipedia approach, we already know that each of the candidates is an event, and are only lacking relevance judgements for tweets, allowing us to have more tweets judged under our 1 minute limit. This allowed us to use our original 30 tweets per event for the Wikipedia approach.

Annotating Tweets

We ran a number of small pilots to test the best method of gathering judgements (i.e. mark relevant, mark non-relevant, or select relevant / non-relevant for each tweet). Empirically we found that all 3 options gave similar results, but that selecting relevant / non-relevant for each tweet seemed to give very slightly more accurate res-

ults when compared to our own annotations, although these differences were mostly found when annotating subjective tweets. However, selecting relevant/non-relevant for each tweet is significantly more work than selecting the relevant or non-relevant tweets only, and fatigues annotators much faster than the other methods. Of the two remaining methods (i.e. selecting only relevant or only non-relevant), we chose to use the selection of non-relevant tweets only as it allows us to more easily use a number of spam detection techniques (described in section 3.2.2.5).

Defining a set of Categories

In additional to gathering relevance judgements, we wanted to gather category information for each of the events. Although the Wikipedia Current Events Portal does assign categories, there is a very large number of categories, each of which is very specific (e.g. 'History', 'Literature', and 'Spirituality' categories). Rather than ask the annotators to choose between a large number of categories, we chose to use the categories defined and used by the Topic Detection and Tracking (TDT) project Allan [2002a]. The 13 categories defined cover a wide range of topics, with a Miscellaneous category for topics which do not fit elsewhere. The full list of TDT categories are shown in Table 3.1.

Quality Control

During our pilot evaluations, we found that a small number of users were abusing the ability to quickly finish evaluations for the detection approaches by answering 'no' to either of the first two questions. To solve this, we implemented a 20 second minimum time limit for all evaluations to deter users who simply wanted to spam submission for easy money. The intuition for this being that users who are only interested quick money will not be willing to wait between successive HITs, and so will not participate in our evaluation.

We also developed several methods of detecting spam submissions so that they can be removed and re-run. We employed a honey-pot technique to detect users who were not correctly classifying tweets as relevant or non-relevant. We insert a tweet from a preselected set of spam tweets which we know do not discuss an event, and since the user has already indicated that the cluster they are evaluating does discuss an event, we can be sure that a spam tweet is non-relevant to the current cluster. If the user does not identify this tweet as being non-relevant then their evaluation is marked as being spoiled and we re-run the evaluation with another user. Of those evaluations sub-

Table 3.1: Combined categories with their corresponding TDT and Wikipedia categories.

Combined	**TDT Categories**	**Wikipedia Categories**
Business & Economy	Financial News	Business, Economics
Law, Politics & Scandals	Elections, Political & Diplomatic Meetings, Legal/Criminal Cases, New Laws, Scandals/Hearings	International relations, Human rights, Law, Crime, Politics, Elections
Science & Technology	Science & Discovery News	Exploration, Innovation, Science, Technology
Arts, Culture & Entertainment	Celebrity & Human Interest News	Arts, Culture, Literature, Religion, Spirituality
Sports	Sports News	Sports
Disasters & Accidents	Accidents, Natural Disaster	Accidents, Disasters
Armed Conflicts & Attacks	Acts of Violence or War	Armed conflicts, Attacks
Miscellaneous	Miscellaneous News	*Anything which is not listed above. e.g. Heath, Transport*

mitted as events, only 999 (4.5% of the 22,114 total evaluations, 286 for the Detection approaches, and 714 for the Wikipedia approach) were marked as being spoiled (i.e., the worker failed to identify the honey-pot), which is a good indication that workers were doing a reasonable job of judging relevance.

In the case of detection approaches, we attempted to find users who were attempting to do as little work as possible. By examining the ratio of evaluations performed to the number of clusters marked as events by that user, we were able to identify users who had submitted either exceptionally high numbers of 'yes' or 'no' answers to the questions *Do the tweets discuss an event?*. We removed users who had performed over 75 evaluations and had over 90% 'yes' answers, or over 90% 'no' answers. This resulted in the removal of 12 users who had performed 4560 evaluations in total. This amounts

to around 35% of the total number of evaluations for the detection approaches. Interestingly, we noted that of the 12 users removed due to spam, 9 appeared in the top 10 users by number of evaluations performed. This suggests that limiting the number of evaluations which a single user can perform could be a very reliable method of reducing noise and spam.

Choosing a stopping point

Whereas tweets from the detection approaches are unranked, tweets obtained using the Wikipedia approach are ranked using the Divergence From Randomness (DFR) [Amati and Van Rijsbergen 2002] retrieval model. This means that as we progress further down the rankings, many of the tweets will start to become non-relevant. Rather than annotate all of the tweets for the Wikipedia approach, we chose to use an incremental approach, inspired by the methodology used by the TDT project, which means that annotation stops once it is unlikely that more relevant tweets will be found.

Possibly need to move this paragraph. Zobel [1998] found that although TREC-style pooling produced test collections which give reliable measures of relative performance, there are potential issues with the depth of the pool used, meaning that many relevant documents are not found (30%-50%), even in smaller collections. They suggest that pool sizes for each topic should be increased when it is highly likely that further relevant documents will be found.

As mentioned in section 3.2.2.2, we split the ranked list of tweets into batches of 30 and had each list annotated 5 times. Once all annotations had been obtained for a single batch, an automatic decision is made based upon the number of tweets where the majority of annotators have marked them as relevant. If a batch contains at least 50% relevant tweets, then the next batch of 30 is suitable for annotation. On the other hand, if a batch contains less than 50% relevant tweets, then annotation is stopped and the event is marked as complete. This process is repeated until all events have been marked as complete.

In order to determine if our stopping point was effective, we created a pilot study where annotators were shown tweets from after the automatic stopping point (i.e. where there were very few or no relevant tweets). Interesting, the number of tweets marked as relevant by annotators was generally very high, often above 50%. We believe that the majority of annotators actually were confused by the lack of relevant tweets, and created their own pseudo-topic based upon the tweets being shown to them. For example, where the event described a mass shooting in Nigeria, all 3 an-

notators seemed to switch to another event, leaving only tweets discussing a bombing at a church in Nigeria as relevant. Although these events share a similar location, the events themselves are very distinct. Unfortunately, because of the short length of each tweet, it is easy for a single term to dominate the rankings – in this case, the only common term between both events was *Nigeria*. This indicates that continuing to ask for annotations after our stopping point would actually harm the accuracy of results, rather than improve them.

Final Detection Evaluations

Annotators was asked to read 13 tweets (selected at random, however kept constant between annotators) from a single candidate event. They were then asked: *"Do the majority of tweets discuss the same real-life topic?"* If they answered 'no' then the evaluation was complete and they were allowed to submit the evaluation. If they answered 'yes', then a further question was posed: *"Do the tweets discuss an event?"* At this point, they were also reminded of our definition of event. Again, annotators who answered 'no' were allowed to submit the evaluation. However, if they answered 'yes' (signalling this the candidate is an true event), then they were asked to re-read the tweets and mark any tweets which did not belong to the event. They were then asked to briefly describe the event and select the category which fits the event. Assuming they had completed all of the above, they were then allowed to submit the evaluation.

Final Wikipedia Evaluation

For each batch, we asked three annotators to read a description of the event, as taken from the Wikipedia Current Events portal. We also supplied a link to a relevant news article for more information about the event if they were still unsure of exactly what had happened or wanted more information. They were then asked to enter a short description of the event, which they might use if they were searching for tweets about the event themselves. Finally, they were asked to read the tweets, marking any non-relevant tweets as so.

3.3 Combining Events from Multiple Sources

Each of the methods produces a different set of events, however these are not disjoint sets in terms of both the events they cover and the tweets they contain. For example, all three approaches independently had an event for the third U.S. Presidential Debate, and unless they are combined, we will have at least three separate 'events' in

our relevance judgements, all of which discuss the same seminal event. Additionally, each method can produce multiple results for the same event. For example, the LSH approach produced no fewer than 40 clusters which discuss the third U.S. Presidential Debate. Although each cluster could potentially be mapped down to specific sub-events within the Debate (such as individual questions or memorable quotes), they would better suit our definition of event if they were combined into a single event (i.e. the Debate as a whole). Given this, we attempt to cluster events together, both from different sources (e.g. the LSH and CS algorithms) and the same sources (i.e. two events from the LSH algorithm), such that they fit our definition of event as closely as possible.

3.3.1 Clustering Features

There are a number of features which could be used for the clustering of events. This section describes the features and the possible issues surrounding their use.

Category Features

Categorization is somewhat problematic for the detection approaches as there was a large amount of disagreement between annotators (discussed in detail in section 3.4). Going back to our example of the third U.S. Presidential Debate, we noted that the LSH approach produced over 40 clusters for the discussing various sub-events and topics within the debate. Many of the subjects of the debate were related to economics, business and international relations. This is an issue because annotators often categorized results based upon the topic of discussion (e.g. New Laws, Political and Diplomatic Meetings), rather than based on the event which caused the debate to take place (the Election). Despite it being clear at a higher level that the debate should be categorized as Election, it is not so clear at the level of sub-events and specific moments within the debate — the categorization changes as we change the level of granularity.

Additionally, because Wikipedia defines its own set of categories, we must create a mapping between the TDT categories and Wikipedia categories if we wish to measure category similarity. Creating a direct mapping between TDT categories and Wikipedia categories would solve the mapping problem but not increase agreement between the detection approaches. Thus, we created a new set of categories, each of which covers a much broader range than either the TDT or Wikipedia categories. Table 3.1 shows the new categories and the corresponding categories from the TDT project and

Wikipedia. These categories greatly increase agreement, which is discussed in detail in section 3.4.2.

Temporal Features

Clearly, temporal features are extremely important in the clustering of events – results which have a significant period of time between them are unlikely to be related to the same seminal event. For example, the three U.S. Presidential Debates are all likely to be very similar in terms of category features, and potentially content-based features. The largest defining factor is the specific time when the events took place.

On the other hand, events which share similar characteristics in terms of both category and content-based features are still relatively common in the same temporal region. Sports events is the best example of this type of event — it is not uncommon for two football matches to occur simultaneously, such as World Cup qualifying matches. These share the same category, the same temporal region, and very similar temporal features, making them particular difficult to distinguish.

We found that by using temporal proximity as a feature for measuring similarity was harmful, and resulted in a very large number of false groupings, a very undesirable property (this is demonstrated in section 3.3.4). Given this, rather than using temporal proximity as an indication of a relationship, we do the converse, and say that events which are temporally dissimilar are unlikely to be related.

Content-based Features

Content based features are some of the most distinguishing features of each result, which can make them difficult to use for event clustering. This problem is most pronounced between results from the same detection approach, where content based features are already used to perform clustering. This means that each of the clusters are generally dissimilar in terms of content, even when discussing the same event, and often contain a very specific set of terms. This makes the use of similarity measures based solely on the content of tweets ineffective when clustering events from the LSH approach. For example, again using the presidential debate as an example, the quote “Well, Governor, we also have fewer horses and bayonets, because the nature of our military’s changed.” was extremely popular and relevant to the debate, however, matching this back to the election is very difficult if the context is not known, and using only tweet content based features to match it to other clusters is very difficult. This means that we cannot rely only on tweet content features when performing

event clustering.

Fortunately, we also have event descriptions written by workers from the crowdsourced evaluation. These tend to be more generic, and seem to offer a more viable method of clustering results together, as they often mention the specific entities involved in an event (such as people and places), and a high level description of the event than cannot be determined using the content of tweets alone.

3.3.2 Clustering Algorithm

For each candidate event e, our algorithm calculates its similarity against every other event e' within a time window T_{time}. The similarity is calculated as so:

$$S_{con} = max(escore(e, e'), dscore(e, e'))$$

$$S_{cat} = cscore(e, e')$$

$$S_{tweet} = tscore(e, e')$$

$$S_{full} = 0.3 * S_{cat} + 0.4 * S_{con} + 0.3 * S_{tweet}$$

where $dscore$ gives the similarity between event descriptions as given by annotators, and $escore$ gives the similarity between named entities extracted from descriptions. $cscore$ gives the similarity between the categories assigned to the event, and $tscore$ gives the similarity between the top 10 most frequent terms in relevant tweets for both events. In every case, cosine similarity is used. The weighting parameters were chosen such that no one feature would be enough to cause a match, reducing the chance of a false match. However, annotator descriptions were given slightly more weight than other features because of their high-level nature and specificity.

If two events are found to have an S_{full} value above threshold T_{sim} then they are clustered together. If both e and e' already have clusters then the two clusters are merged. The pseudo-code for our clustering approach is shown in Algorithm 3.

3.3.3 Event Statistics

Candidates from the detection approaches are considered to be an event if more than 50% of annotators marked it as so and greater than 90% of judged tweets were marked as relevant. This resulted in 382 events for the LSH approach, and 53 for the CS approach. Candidates from the curated approach are considered an event if they pro-

ALGORITHM 3: Event Clustering Approach

```
clustered = ∅;
foreach event e in candidates do
    add e to clustered;
    foreach event e' in candidates but not in clustered do
        S_con = max{escore(e, e'), dscore(e, e')};
        S_cat = cscore(e, e');
        S_tweet = tscore(e, e');
        S_full = 0.3 * S_cat + 0.4 * S_con + 0.3 * S_tweet;
        if S_full ≥ T_sim and time_diff(e, e') ≤ T_time then
            if neither e nor e' in cluster then
                create new cluster containing e and e';
            if both e and e' have clusters then
                merge cluster(e) and cluster(e');
            else
                add e or e' to existing cluster;
            end
        end
    end
end
```

duced at least 1 relevant tweet, resulting in 361 events and 7 non-events (i.e. candidates where no tweets were marked as relevant). In total, this resulted in 796 events before clustering.

Individual tweets are regarded as relevant if more than 50% of annotators agreed. Table 3.2 shows the distribution of tweets broken down by both approach used and type (implicit and explicit). Explicit judgements are those made by human annotators, whereas implicit judgements are tweets from events with high precision (>0.9) which are extremely likely to be relevant, however have not been judged by human annotators. This gave 4,009 explicit and 93,398 implicit judged tweets for the LSH approach, 465 explicit and 15,098 implicit judgements for the CS approach, and 39,980 explicit judgements (with no implicit judgements) for the curated approach. Although the use of implicit judgements will have introduced some noise to the relevance judgements, because we remove candidates with low precision we are able to minimize noise whilst increasing the number of judgements by over 200%.

We then ran the clustering algorithm using a maximum time difference of 6 hours, which was picked specifically to allow for a reasonably lag between the 2 events, whilst still giving a reasonable guarantee that the events generated will fit our definition of event. Categories are assigned to events based on the combined categories defined in Table 3.1. For events where multiple categories are given, the category with the highest frequency was used. In cases where there was a tie between the categories, an

Table 3.2: The distribution of relevance judgements across the different approaches. Explicit judgements are made by human annotators, implicit judgements are taken from events with high precision (>0.9) but not judged by human annotators individually.

Approach	Explicit	Implicit	Total
LSH	4,009	93,398	97,407
CS	465	15,098	15,563
Wikipedia	39,980	0	39,980
Total	**44,454**	**108,496**	**152,950**

author was used to give the deciding vote.

3.3.4 Evaluation of Event Clustering

Since our event clustering approach is unlikely to be perfect and affects the number of events in our relevance judgements, an evaluation of its effectiveness is important. Rather than directly evaluate the results of our clustering approach (i.e. similar to the evaluation of relevant tweets for a cluster), which would have made it difficult to measure any improvements in clustering quality, we wanted to gather relevance judgements which could be used to both measure and improve cluster quality. Amigó et al. [2009] define a number of constraints which need to be met in order for different aspects of cluster quality to be measured. They compared a number of metrics and found only the BCubed metric [Bagga and Baldwin 1998] satisfied all of their constrains. The BCubed metric measures the precision and recall associated with individual items in a distribution, with Recall representing how many items from the same category appear in the target's cluster, and Precision representing how many items in the cluster belong to the same category [Amigó et al. 2009]. Since BCubed averages are calculated over items rather than clusters, this means that we can gather relevance judgements for a subset of the items, rather than the full 796, whilst still being able to accurately estimate the overall quality and choose the best parameters. Additionally, this means that it is not necessary to apply any weighting due to cluster or category size [Amigó et al. 2009].

We took a random sample of 100 items (12.5% of the total), which we refer to as *targets*. For each target, we identified 790 items which had centroid times (i.e. the average time of all the tweets in the item) within a 24 hour window (i.e. 12 hours either side of the target). We call these items *candidates*. For each target, we performed a crowdsourced evaluation. Workers were asked to read a description of the event and

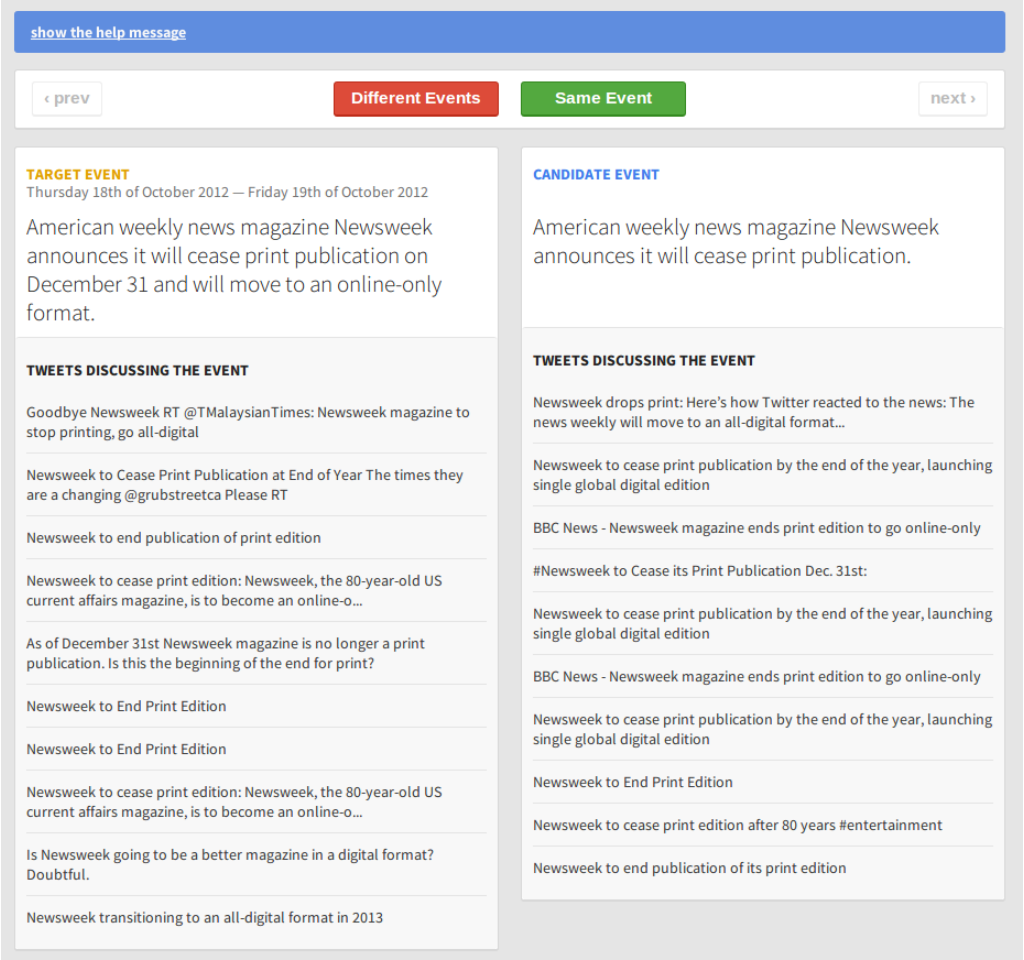

Figure 3.1: A screenshot of the cluster quality evaluation interface used by Mechanical Turk workers.

a random sample of tweets, and for each candidate event, judge if the two events were the same or different. Since many of the targets had a large number of candidates (the highest being 78), we split candidates into batches of 28, reducing the likelihood that workers would become bored or fatigued. Each target event and candidate was evaluated by a minimum of 3 workers. Event descriptions were taken as the longest (by number of characters) description given by users as part of the original evaluation, which we empirically found to be of a high quality.

In order to reduce spam, we again used a honey-pot technique, where a known relevant and a known non-relevant candidate was inserted into the evaluation, for a maximum of 30 candidates per evaluation (28 real candidates, 2 honey-pots). In the case of the known relevant candidate, the worker was simply shown the target event with a different description (the second longest description), and different set of tweets. For the known non-relevant candidate, the worker was shown a candidate which had occurred outside of the candidate window (i.e. more than 12 hours before or after the target). Once again, users who submitted more than 3 bad evaluations (i.e. marking the

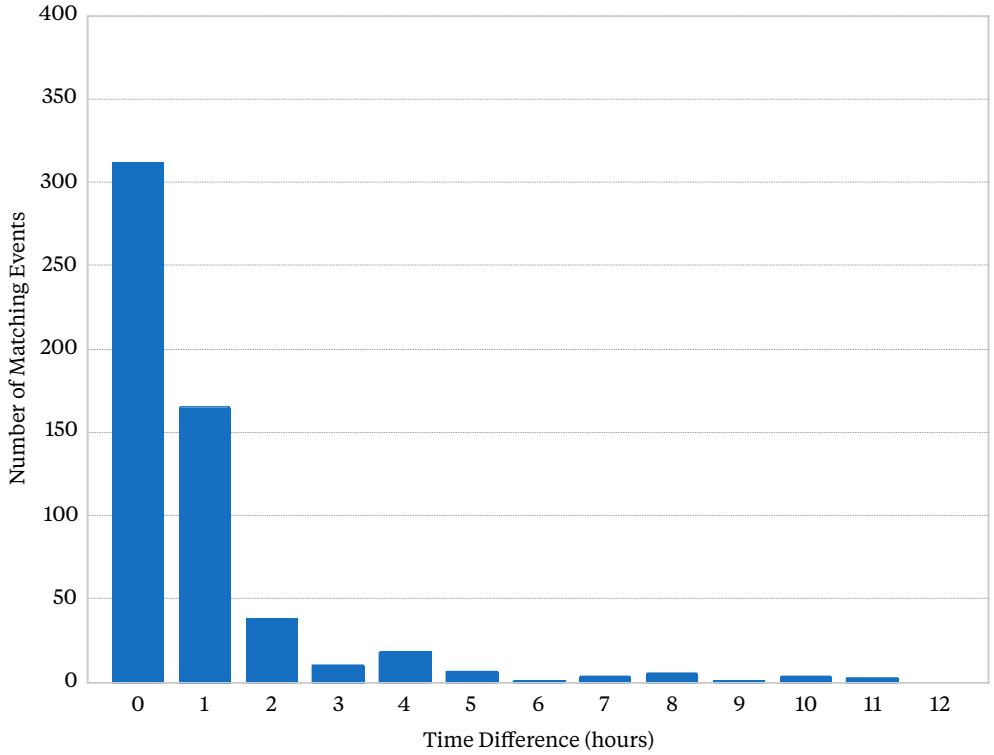

Figure 3.2: The distribution of times between matched events, based upon the number of hours between the centroid times of the events.

known relevant candidate as non-relevant or vice versa) were banned for performing more evaluations for 60 minutes. Candidates were considered relevant when more than 50% of workers identified them as so, which resulted in 235 of the 790 candidates being matched to at least 1 target, giving 589 relevance relationships in total.

Using the parameters proposed in Section 3.3.2 gave the best results, with high average BCubed precision (0.92) and recall (0.86) values. Additionally, our choice of 6 hours appears to be reasonable as only a 3.4% of events were matched outside of a 6 hour window, as show in Figure 3.2.

It is interesting to note that, although we could have used the number of shared tweets as a feature for clustering, it would have made no difference to resulting clusters. Out of 41 cases where events share more than 10 tweets, there is only a single case where they do not have a similarity score above our threshold, and the events are subsequently clustered through shared similarity to a 3rd event. This helps to demonstrate the effectiveness and robustness of our event clustering technique.

3.4 Results & Discussion

In this section, we describe the results of our evaluation. We being by describing the characteristics of the generated corpus, and discuss the effectiveness of the different approaches. We then discuss the annotator agreement, and discuss possible reasons for differences in agreement between the two approaches.

3.4.1 Corpus Characteristics

After clustering, 506 *top-level events* (i.e. events created by combining events from different sources) were produced. In total, 367 events were clustered to create 77 top-level events, and a further 429 top-level events were created using individual events.

As expected, the detection approach seems to closely reflect to the types of events most commonly discussed on Twitter Zhao and Jiang [2011], while the Wikipedia approach gives a more realistic representation of real-world events. As shown in Table 3.3, both approaches produced very different distributions of results, however seem to complement each other, and the combined results show much less variation. For example, the detection approach contributes a large number of *Sports* events, something which is lacking from the Wikipedia approach. Likewise, the Wikipedia approach contributes a large number of events from *Armed Conflicts and Attacks*, and *Business and Economy*, both categories where the detection approach produces less results. This could be due to the volume of discussion associated with each of these topics. *Law, Politics and Scandals* have very few tweets per event in comparison with Sports, meaning that restricting the detection approaches to clusters with at least 30 tweets could have removed many which were discussing these events. Since we did not put this restriction in place for the Wikipedia approach, it would not have Table 3.3 shows the events broken down by category and the type of approach used to generate the event.

The detection method contributed to 186 top-level events, while the Wikipedia approach contributed 342, almost double that of the detection method. However, the detection approaches contribute over 110,000 of the 150,000 relevance judgements in the corpus, with an average of 259 tweets per event cluster. The Wikipedia approach contributes just 39,980 of the relevance judgements, at an average of 85 tweets per event. This is presumably because of the different types of event identified by each of the methods. While the detection approaches rely on the volume of tweets to identify events, the Wikipedia approach does not, meaning that it produces a much

Table 3.3: The distribution of events across the 8 different categories, broken down by method used. The LSH, CS and Wiki columns show numbers of events *before* clustering, while the Clustered column shows the number of events *after* clustering has been performed.

Category	**Clustered**	**LSH**	**CS**	**Wiki**
Armed Conflicts & Attacks	98	3	1	95
Arts, Culture & Entertainment	53	26	3	34
Business & Economy	23	2	1	22
Disasters & Accidents	29	16	4	23
Law, Politics and Scandals	140	124	12	128
Miscellaneous	21	26	6	3
Science and Technology	16	10	2	11
Sports	126	175	24	26

larger number of smaller events. The combination of the two approaches allows their different characteristics to complement each other, producing a much more robust corpus than would have been produced had a single approach been used. If only one approach had been used, then the results would have been unevenly skewed towards either sports or Law and Political discussion, where as the use of both approaches smooths these out to be reflect both the types of event which are happening in the world, as well as what is being discussed on Twitter.

3.4.2 Annotator Agreement

The choice of 5 annotators for the evaluation of the detection approach was useful for a number of reasons, as has already been discussed in section 3.2.2.1, and appears to have been a reasonable choice. Event agreement increases slightly when 5 annotators are used as opposed to 3 ($k = 0.91$ and $k = 0.82$ respectively, using Free-marginal multirater kappa [Randolph et al. 2005]), whilst agreement at a tweet level remains almost unaffected between 5 and 3 annotators ($k = 0.91$ and $k = 0.90$ respectively). Although this suggests that perhaps 3 annotators would have been sufficient for the evaluation of the detection approaches, it would have resulted in many cases where fewer than 3 annotators judged the tweets for a candidate event, breaking the requirements we outlined in section 3.2.2.1.

In the case of the Wikipedia approach, tweet agreement was much lower at 0.72, although this still shows very strong agreement between annotators. It is difficult to say why the agreement is lower for the Wikipedia approach. However, we hypothesizes

that the majority of low quality annotators simply take the easy option for the detection approaches (i.e. answer 'no' to the first question), and never actually judge the tweets. Whereas for the Wikipedia approach, there is no "easy" option, so lazy quality annotators have a detrimental effect on the quality of the results.

Annotator agreement was substantial across the TDT categories ($k = 0.76$). Agreement was further improved when the combined categories were used, giving near-perfect agreement (0.81). This shows that our combined categories not only helped to create a category mapping between the different approaches, but also helped to improve agreement, and thus the categorization of events.

3.5 Conclusion

In this chapter, we described the creation of a new corpus for evaluation of event detection approaches on Twitter. We examined current definitions of event, and proposed a new definition which better fits the characteristics of Twitter. We proposed a methodology for the creation of a large-scale event detection corpora using state-of-the-art event detection approaches, and the Wikipedia Current Events Portal to create a pool of events. We then use crowdsourcing to generate relevance judgements for the pool of events. We then propose a method of merging events from different sources, so that the final events fit our definition of event. We discuss the quality of the results obtained, and note a number of areas which merit further investigation. We make the corpus, which contains 120 million Twitter, and relevance judgements for 150,000 tweets covering more than 500 events, available for further research and development.

CHAPTER 4

Entity-Based Event Detection

Named entities are the building blocks of events; the people, places and organizations involved are crucial in describing an event. For example, given the event "***Hilary Mantel*** *wins the 2012* ***Man Booker Prize*** *for her novel* ***Bring Up the Bodies***", it is clear that named entities (highlighted in bold) play a crucial role in describing an event, and are often enough to decipher what happened. Kumaran and Allan [2004, 2005] showed that the use of named entities can give significant improvements to the performance of event detection techniques on newswire documents [Kumaran and Allan 2005], however their approach cannot cope with the noise of scale of Twitter-sized corpora.

In this chapter, we propose a novel event detection approach which exploits the role that named entities play in describing events. Our approach identifies bursty named entities and performs semantic linking to identify other entities also involved in the event. We evaluate our approach using the collection and relevance judgements created in chapter 3, and show that our approach gives significant increases in detection precision and recall over current state-of-the-art approaches whilst maintaining real-time performance. Parts of this work were published in McMinn and Jose [2015] and McMinn et al. [2014].

4.1 Named Entities in Events

We believe that named entities play a key role in describing events, such as the people involved, or the location where the event took place. Without this information, or some other contextual clue, it is unreasonable to expect a person or machine to determine the specifics of an event. For example, given the document "*A bomb exploded.*", it is impossible to determine who was involved or where the event took place – we are only able to say that a bomb exploded somewhere (assuming the tweet itself is true). Only by introducing entities or other contextual information can we begin to determine the specifics of an event: "***Boko Haram*** *claims responsibility for a bomb which ex-*

ploded in the north-east ***Nigerian*** *town of* ***Potiskum***.". Given this information, we are now able to say who was involved (Boko Haram), where the event took place (Potiskum, Nigeria), and due to Twitter real-time nature, infer with some confidence that the event took place recently.

Clearly entities play an important role in events, and this can be exploited in a number of ways. It is not unreasonable to assume that tweets which do not discuss the same entities (that is to say, do not share at least one common entity) are unlikely to discuss the same event. Conversely, it is not unreasonable to assume that tweets which discuss the same entities are more likely to discuss the same event. We believe that by exploiting the role that entities play in real-world events, we can produce an effective and efficient event detection algorithm. Furthermore, by using the relationship between entities within events, we believe that we can improve upon current state-of-the-art performance using semantic links between entities to improve event-based retrieval.

The use of named entities for event detection has been suggest in the past, Kumaran and Allan [2004] used named entities to improve First Story Detection performance on Topic Detection and Tracking corpora, however there was concern that its effectiveness would be too low when dealing with the low quality and noisy content found on Twitter. Li et al. [2012a] examined the performance of current state-of-the-art Named Entity Recognition (NER) software on Twitter, and although they found that it was possible to significantly improve upon current performance, their results show that out-of-the-box solutions, such as the Stanford NER[1], perform adequately for most tasks.

Choudhury and Breslin [2011] examined how linguistic features and background knowledge could be used to detect specific types of sports events with high precision. However their approach requires significant domain knowledge of the sporting events and large amounts of manual labeling to prepare a classifier for each specific type of event, making it difficult and time-consuming to generalize.

Ritter et al. [2012] used named entities, "event phrases" and temporal expressions to extract a calendar of significant events from Twitter. However, their approach requires that tweets contain temporal resolution phrases, such as "tomorrow" or "Wednesday" for their approach to resolve between an entity and a specific time. This means that smaller events, which do no generate much discussion before they happen, are unlikely to be detected. Additionally unexpected events, which are often the

[1] http://www-nlp.stanford.edu/software/tagger.shtml

events which are of most interest, are unlikely to be detected as they will be discussed as they happen, and without any temporal resolution phrases.

Most similar to our work is that of Li et al. [2017], who use a similar partitioning technique to efficiently cluster documents by splitting the tweet term space into groups they call 'semantic categories', such as named entities, mentions, hashtags, nouns and verbs. To ensure that only novel events are reported, they compare each new event to old events, filtering out any events that have already been reported. They merge events by comparing the top 20% of entity terms, based on term frequency, similar to how we merge events based on entity co-occurrence. They evaluate their approach using the Events 2012 corpus from chapter 3, finding that their approach outperforms the LSH approach when measured using NMI and B-Cubed cluster performance metrics. It is important to note that the work presented in this chapter was first published in McMinn and Jose [2015], 2 years prior to Li et al. [2017].

4.2 Entity-based Event Detection Approach

In this section we describe our entity-based event detection approach. The approach comprises of 6 key stages, as shown in Figure 4.1. Tweets are processed in order using a pipelines architecture that allows for parallel processing, with each component relying only on the output of the previous component to complete its task.

Figure 4.1: The pipeline architecture and components of our entity-based approach.

4.2.1 Tweet Pre-processing

Our approach uses a number of pre-processing steps. This helps to filter out unwanted tweets, such as common types of spam, and attempts to provide a noise free stream of tweets which can used to detect events.

Parsing & Tagging

We use the Stanford CoreNLP Natural Language Toolkit to perform Part of Speech (POS) tagging and Named Entity Recognition (NER) on the text of each tweet. We use

the GATE Twitter POS model [Derczynski et al. 2013] for Part of Speech tagging, and the Stanford 3 class model for Named Entity Recognition.

We extract all nouns, verbs and named entities (persons, locations, and organizations) from each tweet. Nouns and verbs are lemmatized, and entities are kept in their longest form to ensure that names are as distinguishing as possible (i.e. "Paul Ryan" rather than "Paul" and "Ryan"). Different entity classes of the same name are considered to be different. For example, "Spain" in the context of the football team (an organization) is different from "Spain" the location, as this helps to retain as much distinguishing power as possible. We evaluate and give reasoning for this choice in section 4.4.7.1. All terms and entities are converted to lowercase and any non-alphanumeric characters are removed (however whitespace is retained in the case of named entities).

Filtering

Event detection is analogous to a filtering task in many ways – by removing as many non-event related tweets as possible, we are more likely to find event related ones. To this end, we apply a set of filters which remove over 95% of tweets. This has a number of benefits. Firstly, assuming that the filters remove more noise than signal, it becomes considerably easier to extract events. Secondly, unlike other approaches which filter after clustering [Petrović et al. 2010b; Becker et al. 2011b], filtering before clustering reduces the amount of data which needs to be processed, and plays a significant role in the ability of our approach to detect events in a time and space efficient manner.

We first remove tweets that contain no named entities. This is our most aggressive filter, removing over 90% of tweets. As discussed in section 4.1, we believe that named entities play a crucial role in describing events, thus do not believe that this filter significantly harms detection performance, and provide some evidence towards this in section 4.4.7. Furthermore, in order to efficiently cluster tweets, our clustering approach (described in section 4.2.2) requires each tweet to contain at least one named entity, making this filter necessary.

We also remove retweets, which make up approximately 30% of all tweets and are simply a copy of an existing tweet. Retweets require little effort to produce, meaning that they are often associated with the spread of spam, memes and misinformation [Grier et al. 2010], and do not provide any new information. We examine the effect of removing retweets in section 4.4.6, and show that their removal results in a

significant increase in precision.

We also use a range of term-level filters that remove terms and entities that are unlikely to be related to an event or that are known to be associated with spam and noise. As well as stop words and expletives, we remove terms associated with watching television (“watch”, “film”,“movie”, “episode”, etc.) or listening to music (“listen”, “song”, “play”, etc.). This helps to reduce the number of false positives created by large numbers of users watching a television show while using a “second screen” to discuss it, as significant, but fictional events in television shows can often cause reactions that appear similar to real-world events. We also remove terms and entities associated with traditional news and broadcast agencies, such as “bbc news”, “cnn”, “fox news” and “reuters”. Tweets from these agencies often contain their own name, which can cause issues with our entity based clustering and merging approaches (described in sections 4.2.2 and 4.2.5) as the same entity can appear to be associated with a large number of events simultaneously. Finally, terms under 3 characters in length are removed.

4.2.2 Entity-based Online Clustering

Clustering is a commonly used method in event detection to find documents which discuss a similar topic or event, however it is also inherently slow when dealing with large numbers of documents, making traditional clustering infeasible at Twitter scale. A number of solutions have been proposed to solve this, including the use of Locality Sensitive Hashing [Petrović et al. 2010a], which we implement in chapter 3. However, these approaches are general purpose, and do not make use of any domain or task specific information. Given the role that named entities play in describing real-world events, we believe it makes sense to use their presence to improve clustering efficiency and effectiveness for the task of event detection.

Using the premise that tweets discussing an event must contain at least one of the named entities involved in the event, we partition tweets based upon the entities they contain, and add a tweet to a cluster for every entity it contains, as show in in Figure 4.2.

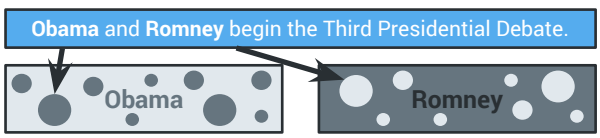

Figure 4.2: Tweets are added to a cluster for each entity the contain. The example shows how a tweet containing both ‘Obama’ and ‘Romney’ would be put into two clusters, one for the entity ‘Obama’ and one for the entity ‘Romney’.

For the purpose of clustering, this can be thought of as having a unique inverted index for every named entity. For each named entity e in tweet d, a list of tweets D is retrieved from the inverted index for e (the inverted index for e does not contain a row for e) and the maximum TF-IDF weight cosine similarity score is calculated between d and each tweet in D.

To ensure that our approach is able to run in real-time, we limit the number of tweets that can be retrieved from an entity's inverted index to a fixed number per term (usually between 100-1000), and use only the top 10 TF-IDF weighed terms per tweet. Less than 1% of tweets from our test collection contain more than 10 terms, meaning that we lose very little information by enforcing this limit, whilst ensuring an upper bound of $10N$ comparisons are made, where N is the maximum number of documents retrieved per term. Nouns, verbs and named entities are used when calculating the cosine similarity between tweets, however the named entity whose inverted index was used to retrieve comparison documents is given a weight of 0 in order to give as much weight as possible to other topical terms.

If the maximum score is above a set threshold (usually in the range $0.4 - 0.6$ [Petrović et al. 2010a], discussed in section 4.4.4), then d is added to the same cluster as its nearest neighbour. If the nearest neighbour does not already belong to a cluster, then a new cluster is created containing both tweets and assigned to entity e. The new tweet is then added to the inverted index for entity e.

It should be noted that unlike traditional event detection approaches, we do not assume that the formation of a new cluster indicates a new event. Algorithm 4 shows the pseudocode for our entity based cluster approach.

ALGORITHM 4: Entity-based method of clustering

```
1  foreach tweet d in corpus do
2      foreach entity e in d do
3          foreach term t in d do
4              foreach tweet d' in index_e that contains t do
5                  update score(d, d');
6              end
7          end
8          score_max(d) = max_d' {score(d, d')};
9          if score_max(d) >= threshold then
10             add d to cluster(d');
11         end
12         add d to index_e;
13     end
14 end
```

4.2.3 Identifying Bursty Entities

For an effective event detection approach, it is important to have a method of detecting significant events and filtering the mundane. We do this by looking for temporal bursts in the frequency of an entity, which can occur over periods ranging from a few minutes to several hours. To model this, we use a set of windows for each entity to capture their frequency over time, starting at a 5 minutes, and doubling in length up to 320 minutes (i.e. 5, 10, 20, ..., 320).

We use the Three Sigma Rule as the basis for a light-weight burst detection approach, which states that a value is considered to be practically impossible if it is further than 3 standard deviations from the expected value [Pukelsheim 1994]. For the windows, we maintain mean and standard deviation values, updating them periodically with the current entity frequency. It is possible to efficiently compute moving mean (μ) and standard deviation (σ) values using a set of three power sums s_0, s_1 and s_2, where s_j, μ and σ at time period $n+1$ for data series x is shown below:

$$s_{jn+1} = s_{jn} + x_{n+1}^{j}$$

The current mean and standard deviation values (μ and σ respectively), can then calculated as so:

$$\mu = \frac{s_1}{s_0} \qquad \sigma = \sqrt{\frac{s_0 s_2 - s_1^2}{s_0}}$$

The μ and σ values for each window are updated periodically based upon the length of the window (i.e., a 5 minute window is updated every 5 minutes). In our evaluations, we find that using full historic frequency information for each window (section 4.4.3), and instead use only the past 12 values to calculate mean and standard deviations.

When a tweet which is no longer covered by the largest window it is removed from all inverted indexes. This helps to reduce the number of comparisons required when calculating the nearest neighbour, and allows resources to be freed for incoming tweets. We do not believe that removing older tweets will affect the effectiveness of the algorithm due to the real-time nature of Twitter, as tweets which are more than a few hours old are generally not relevant to any ongoing real-world events, and those which are will hopefully have already been marked as so by the algorithm, ensuring that they are recorded elsewhere.

Once a tweet has been clustered and added to entity indexes (as shown in Algorithm 4), we update the entity statistics, and check if the newly added tweet caused the en-

tity to burst in any of the windows. If the number of tweets in a given window is greater than $\mu + (3 \cdot \sigma)$, then we say that the given window is bursty, and the entity is part of a new event.

In order to smooth and reduce noise, statistics are not updated while a window is bursting, and windows are kept in a bursting state for $1.5 \times window\_length$ after the window's statistics suggest that it has stopped bursting. For example, an entity which is marked as bursty in the 80 minute window would remain bursting for 120 minutes after the 80 minute window stops bursting. The extra time allows for fluctuations in frequency as an event develops, and helps prevent a state of flux where an entity rapidly switches between bursting and normal states.

4.2.4 Event Creation and Cluster Selection

Once a burst has been detected, an event is created and associated with the bursting entity for the duration of the burst (we address how an event could be associated with multiple entities in section 4.2.5). However, the event does not yet have any tweets associated with it. We cannot simply take all of the tweets posted during the burst as these will contain background topics about an entity, such as discussion about visiting a location or listening to a particular artist's music. To solve this, we propose the use of a number of heuristics to select significant clusters which are the most likely to be related to the event that caused the burst.

We require that the centroid time of a cluster (i.e. the average timestamp associated with all tweets in a cluster) is greater than B_e, where B_e is the time at which the entity began to burst. This helps to ensure that clusters which discuss background topics are not included as they are likely to have existed for some time before the burst took place. A cluster's centroid time is updated as new tweets are added, ensuring that clusters which initially had a centroid time prior to the burst can still be added to an event. This is important as it allows clusters containing early reports of the event, which often occur before any burst takes places, to be included.

We also require that a cluster meets a minimum size threshold, usually between 5 and 20 tweets (10 is used for our evaluations). This is to ensure that the cluster covers a significant portion of the event and to prevent small but noisy clusters from being included. The minimum size has a large effect on the precision of our approach, and large minimum cluster sizes can give a significant increase in precision for a small reduction in recall. We discuss the minimum cluster size in detail in section 4.4.4.

An event is kept alive as long as it has at least one bursting entity associated with it. Once all entities associated with an event have stopped bursting, the event is finalized, and no more clusters or tweets can be added to it.

4.2.5 Event Merging and Entity Linking

New events are related to only a single entity. However, most events involve more than one entity, such as a person and a location, or an interaction between two organizations. In additional, events are often discussed using a number of synonyms. For example "Barack Obama" is often shorted to simply "Obama". These issues mean that there are generally a number of Event objects, each created by separate named entities but of which discuss the same real-world event. To solve this, we create links between events which are related to the same real-world event using entity co-occurrence statistics. If an entity is mentioned in at least 50% of tweets in an event, then we say there is a potential link between the event and the entity.

If such a potential link is found, a number of checks are performed to ensure that the entity is viable and likely to be related to the event. Firstly, we check that the potential entity is bursting and therefore is already associated with an event. Secondly, we verify that the entity has not already been associated with the event. This requirement prevents long-running events from consuming popular or frequently bursting entities. For example, an event which is associated with both "Obama" and "Romney" would not automatically be merged with 'Romney' should it stop bursting and then burst again before the original event had finished. Instead, the Romney entity would form a new event, which may then link itself to the previous event if Obama occurs in more than 50% of tweets.

If event E_1 finds a potential link with entity e, and entity e passes both of the above checks, then event E_1 is merged with the event belonging to e. This merging process can happen any number of times to produce events with many entities, and consisting of all the clusters and tweets from both. Entity frequencies are then recalculated and any new semantic links are then followed to merge yet more events.

Once an entity has stopped bursting, it is no longer linked with an Event and no new clusters from that entity are added. However, clusters for an entity which have already been added are still updated, and any new tweets are reflected in the entity frequencies for the event. Events are kept active until they contain no more active entities and all clusters in the event have become inactive (i.e. all the of the tweets in the cluster have been removed from the inverted index associated with its entity). This helps to

capture more information about the event, even after discussion has died down.

Entity Normalization

As described in section 4.2.2, entities are kept in their longest form rather than split into individual components (e.g. 'Barack Obama', rather than 'Barack' and 'Obama'). However, this can cause issues as multiple events are created for different forms of the same entity. Using our previous example, it is unlikely that single tweet will mention both 'Barck Obama' and 'Obama', resulting in two separate events that our entity linking approach is unlikely to merge.

To solve this, we perform a naive entity disambiguation technique. When measuring the frequency of entity mentions within an event, we perform a normalization step, which counts the last term of people and organization names as a separate entity. For example, for event mention of 'barack obama', we also increase the count of 'obama'. Note that we only do this for People and Organizations, but not Locations, as locations tend to lose much of their meaning when split (e.g. United Kingdom Kingdom).

4.3 Experimentation

In order to evaluate our approach, we make use of the Events 2012 collection we created in chapter 3. Although the collection contains 150,000 relevance judgements for 506 events, we note that it does not cover all events that happened during the 28 days it covers, or even that all tweets relevant to events it does cover have been identified. Whilst this does not prevent us from using the collection for comparison purposes, it means that event precision and recall values can only be estimated. Whilst this is an issue in many IR collections, we note that the effect is more pronounced when dealing with event detection as, unlike most IR tasks, there is no query. This makes it likely that we will detect events that are not in the judgements. We verify this hypothesis and show that we detect a large number of events which are not in the relevance judgements by performing a crowdsourced evaluation, described in sections 4.3.1 and 4.4.

We ran our entity-based approach on the collection, treating it as a stream of tweets ordered by creation time. Named Entities and Part of Speech Tagging was performed using the Stanford POS Tagger[2] 3.2.0. Unless otherwise stated, all runs were performed using the top 10 IDF-weighted terms per tweet, a maximum of 500 tweets

[2] http://www-nlp.stanford.edu/software/tagger.shtml

were retrieved per term, and a minimum cosine similarity of 0.5. Evaluations were performed on all events with more than 30 tweets, using a 5% threshold and/or minimum of 15 matching tweets, with full descriptions of these thresholds and the rationale for their choices described in section 4.5. Any variation from these thresholds is noted alongside the results presented in section 4.4.

We believe that results presented here, when first published in 2015, were the first in-depth and comparable evaluation of an event detection approach for Twitter.

4.3.1 Crowdsourced Evaluation

Given that no event detection technique for Twitter has been robustly evaluated against a publicly available Twitter collection, the only options available to us as a baseline are the LSH [Petrović et al. 2010a] and CS [Aggarwal and Subbian 2012] approaches used to generate the collection in chapter 3. This means that the results are extremely biased towards these approaches, and only a crowdsourced evaluation will allow for a direct comparison between our entity-based approach and the baselines.

As we use the results from the crowdsourced evaluation, baseline parameters have not changed from those used in chapter 3, however for completeness we include them here. The LSH approach used 13 bits per key, 70 hash tables, and a minimum cosine similarity values of 0.55. Events were extracted every 100,000 tweets, and only clusters with entropy values inside the optimal range (3.5 - 4.25) were used [Petrović et al. 2010a]. The CS approach was run with an input size of 1200 clusters. An α value of 12 was used to prevent slow growing clusters from being included in the results [Aggarwal and Subbian 2012]. Neither baseline used retweets as empirical manual investigations before the crowdsourced evaluation found them to have a significant negative impact on precision.

In order to keep the comparison between the baselines and our approach as fair as possible, we use the same methodology here as we did in chapter 3 to gather relevance judgements for the baseline approaches.

Since resources were limited, and our approach generates a large number of candidate events, we use a random sample of 250 events from the 1210 candidate events identified by our approach. For each event, we asked 5 annotators to answer a questionnaire. Each annotator was asked to read 13 tweets (selected at random however kept constant between evaluations) from a single candidate event. They were then asked: 'Do the majority of tweets discuss the same real-life topic?'. If they responded 'no'

then the evaluation was complete and they were allowed to submit the evaluation. If they answered 'yes', then a further question was posed: 'Do the tweets discuss an event?'. At this point, they were also reminded of the our definition of event, as given in chapter 3. Again, annotators who answered "no" were allowed to submit the evaluation. However, if they answered "yes" (signalling that the candidate is an event), then they were asked to re-read the tweets and mark any non-relevant tweets as so. They were then asked to briefly describe the event and select the category which best fit the event. They were allowed to submit the evaluation. Candidate events were considered to be an event if the majority (3/5) of annotators marked them as so.

To keep category classification fair, we used the same 13 categories defined by the Topic Detection and Tracking project [Allan 2002a] as they were used to evaluate events produced by the two baseline approaches in chapter 3. We mapped the TDT categories back to the categories used by the collection. A number of spam and quality control measures were used, identical to those described in chapter 3.

4.4 Results

Table 4.1 shows results for the two baseline approaches and our entity-based approach when run and evaluated against the Events 2012 collection. We give results for both a crowdsourced (Crowd) evaluation and an automatic (Auto) evaluation for our entity-based approach. We present results for our best performing automatic evaluation, which filtered out events with fewer than 100 tweets but used otherwise default parameters as described in section 4.2.

Automatic evaluation gives our entity-based approach an estimated Precision of 0.348 and recall of 0.292 for events with 100 or more tweets. This gives the highest F1 measure (0.306) across all automatic runs tested. Precision beats that of the LSH baseline significantly, despite being disadvantaged due to the partial relevance judgements.

Table 4.1: Results for 2 baseline approaches (LSH & CS), as well as crowdsourced and automatic results for out entity-based approach using the Events 2012 collection. Automatic evaluation was carried out on events with 100 or more tweets, whilst the crowdsourced evaluation was carried out on 250 events with 30 or more tweets and scaled to the full 1210 events.

	CS	LSH	Entity (Crowd)	Entity (Auto)
Precision	53/1097 (0.048)	382/1340 (0.285)	769/1210 (0.636)	162/465 (0.348)
Recall	32/506 (0.063)	156/506 (0.308)	194/506 (0.383)	148/506 (0.292)
F1	0.054	0.296	0.478	0.306

Recall is slightly lower than that of the LSH approach, although by only a small margin (0.016). At a minimum event size of 75, automatic evaluation shows that our entity-based approach outperforms the LSH approach in both precision (0.302) and recall (0.310). A detailed investigation of performance across a range of minimum event sizes is given in section 4.4.2

For the crowdsourced evaluation, events with fewer than 30 tweets were removed so that results were directly comparable to the baseline approaches. Precision values from the 250 crowdsourced evaluations were used to extrapolate how many of the 1210 candidate events were likely to be true events. Of the 250 candidate events, 159 were identified as true events by the majority of annotators, giving a precision of 0.636. Scaled up to include all events, this means that approximately 769 of the 1210 candidate events are true events. Recall for the crowdsourced run was calculated automatically using events with 30 or more tweets. We discuss the difficulties of automatic evaluation and differences between automatic and crowdsourced evaluation in more detail in section 4.5.

4.4.1 Category Breakdown

Table 4.2 shows recall across the eight categories defined by the Events 2012 collection using a minimum event size of 30 tweets. There is a large variation in recall across the categories. Our approach seems to be most effective at detecting events categorized as 'Armed Conflicts & Attacks' ($R = 0.520$) and 'Disasters & Accidents' ($R = 0.448$). This is extremely promising as these are the types of event that are most likely to benefit from eye-witness accounts and the use of social media as the event unfolds. The ability to find information about these types of event in real-time can be useful for law enforcement and emergency services, and serves as one of main motivations for event detection and tracking on Twitter.

Our approach also appears to effective at detecting events in the 'Business & Economy' ($R = 0.391$), 'Sports' ($R = 0.373$), and 'Law, Politics & Scandals' ($R = 0.386$) categories. Law, Politics & Scandals, as well as the Sports events make up over 50% of the total events in the collection, so given our approach's overall high recall, it is not surprising to find that it performs well on events in these categories. This is most likely due to a number of factors. Firstly, these types of event tend to focus on a small number of easily identified entities, such as sports teams, politicians, or company names. Secondly, these types of event are of interest to a large number of people, making them more likely to burst and be detectable, with sports events in particular

being well suited to discussion on social media, something we examine later in this section.

Our approach performs worst on 'Miscellaneous' ($R = 0.190$), 'Arts, Culture & Entertainment' ($R = 0.226$), and 'Science & Technology' ($R = 0.250$) events. The low recall for science and technology events can be somewhat explained by a lack of easily detectable named entities, particularity for science events, such as "Astronomers detect what appears to be light from the first stars in the universe". Of the 21 Miscellaneous events, 10 of them have fewer than 15 tweets in the relevance judgements which contain named entities. This lack of named entities makes miscellaneous very difficult to detect for our approach, and the effect is examined in detail in section 4.4.7. Low recall entertainment events could be explained by our removal of tweets that contain terms related to television, such as 'watch', as many of the events are broadcasts of award shows or the launch of new television shows.

Table 4.2: Distribution of detected events across the 8 categories defined by the collection. The average entities shows the average number of entities linked to each event for the specific category.

Category	**Recall**	**Average Entities**
Armed Conflicts & Attacks	51/99 (0.520)	1.85
Arts, Culture & Entertainment	12/54 (0.226)	1.68
Business & Economy	9/24 (0.391)	3.66
Disasters & Accidents	13/30 (0.448)	2.65
Law, Politics & Scandals	54/141 (0.386)	3.35
Miscellaneous	4/22 (0.190)	1.78
Science & Technology	4/17 (0.250)	2.91
Sports	47/127 (0.373)	4.01

The Average Entities column of Table 4.2 shows the average number of entities per detected event for each category. There is a moderate positive correlation between the average number of entities per detected event and category recall ($r = 0.52$). This is not unexpected, as the more entities that are involved in an event, the easier it is for our entity-based approach to find tweets and other content related to it.

Sports events have, on average, the most entities per event. This makes sense given that sports events are generally team based or involve a large number of people. The large number of entities per sports event also suggests that our approach is reasonably successful at finding and linking entities which discuss the same events. Empirical evaluation of the entity linking seems to show this to be the case. For example, our

approach produced an event with the following entities for a football match between Manchester United and Stoke City on 20th October, 2012: *"carrick, hernandez, kagawa, evans, nani, de gea, cleverley, buttner, rooney, rafael, fletcher"*. Each of the entities refers to one of the Manchester United players that day, and whilst an ideal list of entities would also have included Stoke City players, this demonstrates how effective our approach can be at creating semantic links between entities involved in an event.

Miscellaneous events are by their very nature difficult to accurately classify, which makes analysing their relatively low performance more difficult. Empirical examination of the 22 Miscellaneous events in the collection shows that many of them are simply not very bursty or are developments in long-running events, such as a call by the U.N. to release Iranian lawyer Mohammad Ali Dadkhah who was imprisoned six months earlier. Events such are these are particularity hard for our approach to detect because the volume of discussion is relatively low, and new developments tend not to generate wide spread and bursty discussion when compared to novel events.

4.4.2 Event Size

Table 4.3 shows how the effectiveness of our entity-based approach varies as the minimum event size is increased from 30 tweets to 300. Overall effectiveness, taken as the F1 score, increases between minimum sizes of 30 and 100 tweets, reaching a maximum F1 score of 0.318. Overall effectiveness decreases as the minimum event size is increase above 100 as increases to precision are outweighed by decreases in recall.

Table 4.3: Effectiveness of our entity-based approach at varies minimum event sizes.

Min. Tweets	Precision	Recall	F1
30	242/1210 (0.200)	194/506 (0.383)	0.263
50	199/799 (0.249)	173/506 (0.342)	0.288
75	176/582 (0.302)	157/506 (0.310)	0.306
100	162/465 (0.348)	148/506 (0.292)	0.318
150	131/329 (0.398)	124/506 (0.245)	0.303
300	85/178 (0.478)	91/506 (0.180)	0.261

4.4.3 Burst Detection

Our burst detection technique is one of the key components in our detection pipeline. Events begin and end based on our burst detection technique, so it is important that we examine different aspects of how we have configured the burst detection approach.

Table 4.4: Effectiveness of our approach with different numbers of windows.

Windows	Precision	Recall	F1
30 Tweets			
6	225/1040 (0.216)	185/506 (0.366)	0.272
7	242/1210 (0.200)	194/506 (0.383)	0.263
8	248/1340 (0.185)	198/506 (0.391)	0.251
100 Tweets			
6	141/374 (0.377)	126/506 (0.249)	0.300
7	162/465 (0.348)	148/506 (0.292)	0.318
8	175/555 (0.315)	157/506 (0.310)	0.313
300 Tweets			
6	66/136 (0.485)	74/506 (0.146)	0.225
7	85/178 (0.478)	91/506 (0.180)	0.261
8	101/235 (0.430)	101/506 (0.200)	0.273

Table 4.4 shows how decreasing or increasing number of windows affects the performance of our approach at a number of different minimum event sizes. Decreasing the number of windows to 6, meaning that windows cover period from 5 minutes up to 160 minutes, results in a slight increase in precision across all event sizes, at the cost of recall. The effect is mirrored by increasing the number of windows to 8, so that the maximum period is 640 minutes: recall increases but precision decreases. These differences appear to be linked primarily to the number of events returned: as the number of candidate events increases, recall increases but precision decreases; a pattern that is common across a wide range of IR tasks and models.

Our default of 7 windows appears to offer the best ratio of precision and recall using a starting window length of 5 minutes, and gives the highest overall F1 measure of 0.318 at a minimum event size of 100 tweets. We note, however, that at 30 tweets, 6 windows gives a higher F1 score than 7 windows: although 6 windows detects only 9 fewer events, it producing 170 fewer candidate events. This could be caused by the reduced maximum burst length with 6 windows causing events to be ended earlier, and so fewer reach the minimum size of 30.

Table 4.5 shows how limiting the amount of historic data used to calculate mean and standard deviation values for burst detection affects the overall performance. Although the performance differences between 6 and 24 periods is small, and there does not appear to be a single best history length across the range of event sizes, there is a

Table 4.5: The effect of using only data from the last N updates when calculating mean and standard deviation values.

History Length	Precision	Recall	F1
30 Tweets			
No Limit	241/1293 (0.186)	190/506 (0.375)	0.249
6 Periods	237/1139 (0.208)	192/506 (0.379)	0.269
12 Periods	242/1210 (0.200)	194/506 (0.383)	0.263
24 Periods	247/1262 (0.196)	191/506 (0.377)	0.258
100 Tweets			
No Limit	154/476 (0.324)	137/506 (0.271)	0.295
6 Periods	159/459 (0.346)	146/506 (0.289)	0.315
12 Periods	162/465 (0.348)	148/506 (0.292)	0.318
24 Periods	163/465 (0.351)	142/506 (0.281)	0.312
300 Tweets			
No Limit	82/174 (0.471)	81/506 (0.160)	0.239
6 Periods	89/174 (0.511)	97/506 (0.192)	0.279
12 Periods	85/178 (0.478)	91/506 (0.180)	0.261
24 Periods	89/179 (0.497)	86/506 (0.170)	0.253

clear and significant difference between limited and unlimited historical data.

By limiting the amount of historic data used at each window size, we remove a type of smoothing. Placing no limits on historic data means that we capture the change in usage of each entity, but also changes in how the entity is used through the day as people wake up, go to work, return home, and go to sleep. The overall volume of tweets changes as the day progresses, introducing variance that affects the standard deviation. No limits of historic data also means that events which build slowly, such as discussion of an upcoming sports event or political debate can cause a 'burst' despite being gradual increases in volume. By limiting the amount of historic data used, we ensure that the statistics reflect how the entity is discussed recently, not historically. This might appear counter-intuitive, as the aim of the burst detection is to detect recent changes in the volume of discussion around an entity. However, our use of multiple window lengths means that our burst detection approach captures information over a wide range of time periods. Assuming 7 windows, with the smallest period of 5 minutes, and using only 6 historic values, our shortest window uses data from the past $6 \times 5 = 30$ minutes, whilst our longest window covers a period of $6 \times 320 = 1920$ minutes, or 32 hours.

Table 4.6: Effects of minimum similarity thresholds on detection performance.

Min. Similarity	Precision	Recall	F1
	30 Tweets		
0.40	256/1430 (0.179)	206/506 (0.407)	0.249
0.45	242/1210 (0.200)	194/506 (0.383)	0.263
0.50	227/1133 (0.200)	180/506 (0.356)	0.256
0.55	205/929 (0.221)	175/506 (0.346)	0.269
0.60	181/804 (0.225)	154/506 (0.304)	0.259
	100 Tweets		
0.40	176/566 (0.311)	155/506 (0.306)	0.309
0.45	162/465 (0.348)	148/506 (0.292)	0.318
0.50	154/446 (0.345)	135/506 (0.267)	0.301
0.55	134/346 (0.387)	128/506 (0.253)	0.306
0.60	113/300 (0.377)	111/506 (0.219)	0.277

4.4.4 Clustering

Table 4.6 shows how different similarity thresholds affect the performance across a range of event sizes. High thresholds make it more difficult to cluster similar tweets, meaning that there tend to be a larger number of smaller clusters, as well as more tweets that belong to no cluster. Since our approach only adds clusters above certain sizes to an event (see Table 4.7 for details), a decrease in the average size of a clusters means that fewer events are discovered. Conversely, a low threshold makes it easier to cluster tweet, resulting in more large clusters. This means more clusters can be added to events, and results in more events overall. This is reflected by the number of events at varying levels of similarity and across the two minimum event sizes.

Table 4.7 shows the effect of different minimum cluster sizes on the effectiveness of our approach. Clusters are only added to an event if they contain more the minimum number of tweets, so the minimum cluster size has a significant effect on the overall performance, particularity at lower minimum event sizes. The effect on overall performance (F1 score) is less pronounced for larger events as these events have much higher volumes of discussion, so the number of large clusters that can to be added to the event is much higher.

Increasing the minimum cluster size has a significant effect on precision at both minimum event sizes. There is a distinct jump in precision when the minimum cluster size is increased from 5 to 10 tweets, but with only a very small impact on recall, mo-

tivating our use of 10 as the minimum cluster size in all of our evaluations.

Table 4.7: Effects of minimum cluster size on detection performance.

Min. Tweets	Precision	Recall	F1
30 Tweets			
2 tweets	306/2575 (0.119)	227/506 (0.449)	0.188
3 tweets	292/2038 (0.143)	220/506 (0.435)	0.216
5 tweets	261/1644 (0.159)	207/506 (0.409)	0.229
10 tweets	242/1210 (0.200)	194/506 (0.383)	0.263
20 tweets	196/816 (0.240)	170/506 (0.336)	0.280
100 Tweets			
2 tweets	216/1032 (0.209)	181/506 (0.358)	0.264
3 tweets	192/795 (0.242)	167/506 (0.330)	0.279
5 tweets	171/621 (0.275)	153/506 (0.302)	0.288
10 tweets	162/465 (0.348)	148/506 (0.292)	0.318
20 tweets	129/326 (0.396)	121/506 (0.239)	0.298

4.4.5 Nouns, Verbs and Hashtags

Table 4.8 shows the effect of using different term combinations for clustering. Note that for verb only clustering, although nouns were not used when calculating similarity scores, the use of named entities to partition documents means that proper nouns were still used for clustering in an ad-hoc manner. Despite this, we feel that results presented here are still interesting and insightful.

Although F1 scores show only small changes, both precision and recall values seem to be greatly affected by type of terms used for clustering. The use of nouns only gives the highest recall but the lowest precision ($F1 = 0.249$), whereas using verbs only results in the lowest recall but the highest precision ($F1 = 0.259$). Using both nouns and verbs seems to take best the characteristic of both, giving the highest overall F1 score ($F1 = 0.263$. The high recall associated with nouns fits with our hypothesis that events are about entities, as named entities are proper nouns, and entity classes (i.e. city, person, plant) are common nouns. If nouns had not been used to describe these events then we would not have been able to detect them. This is again reflected in the low recall when using verbs only, and had we been able to remove the dependency on named entities (i.e. proper nouns) then recall would have been much lower. At the highest level, these results seem at agree with our premise that events describe the effect of a verb on a noun (a real-world entity).

Table 4.8: The effect of using different combinations of nouns (NN), verbs (VB) and hashtags (HT) as terms for clustering on events with at least 30 and 100 tweets.

POS	Precision	Recall	F1
30 Tweets			
NN Only	242/1324 (0.183)	198/506 (0.391)	0.249
VB Only	196/912 (0.215)	165/506 (0.326)	0.259
NN, VB	242/1210 (0.200)	194/506 (0.383)	0.263
NN, VB, HT	232/1174 (0.198)	192/506 (0.379)	0.260
100 Tweets			
NN Only	156/522 (0.299)	146/506 (0.289)	0.294
VB Only	122/367 (0.332)	113/506 (0.223)	0.267
NN, VB	162/465 (0.348)	148/506 (0.292)	0.318
HT, NN, VB	157/447 (0.351)	142/506 (0.281)	0.312

The use of Hashtags seems to cause a small reduction in both precision and recall, a somewhat unexpected result, as Hashtags are commonly thought to be very good indicators for the topic of a tweet. We hypothesis that this is due to the specificity of named entities, and that by requiring every tweet to contain a named entity we are removing any topical uncertainty and rendering Hashtags redundant as an indicator of topic.

4.4.6 Retweets

The use of retweets has a significant impact, reducing precision from 0.200 to 0.063. The use of retweets does provide a small, but insignificant increase in recall (from 0.383 to 0.390), and can likely be attributed to a 60% increase in the average number of tweets per event from 125 to 198, creating many events with more than 30 tweets. This finding is somewhat unsurprising as retweets are commonly associated with the spread of spam and require little effort to produce.

4.4.7 Named Entities

One concerns with our entity-based approach is the use of entities on event and tweet recall, since we rely on entities to cluster tweets and detect events. Running the Stanford POS Tagger and NER over tweets from the relevance judgements shows that 47.4% of relevant tweets contain at least one entity. This is promising, and considerably

higher than the 11% of tweets that contain name entities across the collection as a whole, confirming our hypothesis that there is a relationship between entities and events. Our approach achieves a tweet recall of 0.242 across the events it detects, and a recall of 0.511 if we measure only against tweets in the relevance judgements which contain a named entity.

This does highlight one drawback to our entity based approach. Even if we were to detect every event in the collection, we could never achieve a tweet recall above 0.474. Some of this is likely due to the difficulty of NER on Twitter, as noted by Li et al. [2012a], and could be improved with better NER models for Twitter. However, it is likely that the majority of tweets simply do not contain any named entities, meaning that we must consider the effect this has on detection effectiveness – if an event has very few or no tweets with named entities then our approach will be unable to detect them.

Of the 506 events in the relevance judgements, 14 have fewer than 5 associated tweets, 42 have fewer than 15, and 72 have fewer than 30. In addition, 41 events in the relevance judgements have fewer than 5 tweets with entities, 109 events have fewer than 15, and 163 have fewer than 30. For those 41 events with fewer than 5 tweets containing entities, even if our system was to perform perfectly, we would be unable to detect them – accounting for just over 8% of all the events in the collection.

Entity Classes

Table 4.9: The precision and recall as the minimum event sizes is increased.

Entity Classes	Precision	Recall	F1
30 Tweets			
3 Classes	242/1210 (0.200)	194/506 (0.383)	0.263
Single Class	257/1389 (0.185)	201/506 (0.397)	0.252
100 Tweets			
3 Classes	162/465 (0.348)	148/506 (0.292)	0.318
Single Class	170/525 (0.324)	152/506 (0.300)	0.312
300 Tweets			
3 Classes	85/178 (0.478)	91/506 (0.180)	0.261
Single Class	94/193 (0.487)	91/506 (0.180)	0.263

There are a number of cases where entities with the same name, such as Spain the football team and Spain the land mass are different classes of entity. In this case,

Spain can be both an Organization (the football team) and a location (Spain the land mass). Table 4.9 shows the effect of differentiating between classes, and ignoring classes. Using a single class seems to give a small increase in recall at the cost of a small decrease in precision when compared to using multiple classes. However, when measured using F1, the multiple entity classes has a small performance advantage up to events containing at least 100 tweets, whereas the single entity class seems to perform slight better on larger events. This increase in recall using a single class could be caused by the burst detection having more data to work with per entity – rather than tweets being partitioned into 3 separate types, they are left as one, making bursts easier to detect. However, closer inspection of the events detected shows that is unlikely to be the only cause. We find that using a single class tends to merge events which are about different entities with the same name. For example, a football match between Spain and France takes place at the same time as the captain of the Liberian-owned Prestige oil tanker goes on trial in Spain. Because the single entity class is unable to differentiate between the Spanish football team and Spain the location, then the 2 events cannot be distinguished from each other, producing a single large event with an effective recall of 2 events. Despite the fact that only a small percentage of the tweets are about the trail, they still cover over 65% (25/38) of the tweets in the relevant judgements for the trail, so it is difficult to simply say that these small events have not been detected, even when merged with other larger events. There are a number of examples where similar errors are made, and this appears to be the main cause for the increase in recall and the higher number of of large events (193 with at least 300 tweets in comparison with only 178 when using multiple classes). Section 4.5 further discusses issues with automatic evaluation of event detection approaches and suggests how some of these challenges can be overcome.

4.5 Event Detection Evaluation Approaches

Our work to build the Events 2012 collection remains the only publicly available twitter corpus with a ground truth for evaluating event detection approaches, and very few detection approaches have been thoroughly evaluated on a publicly available dataset. To this end, we discuss a a number of challenges associated with evaluating event detection approaches using the Events 2012 collection, and examine how our automatic evaluation approach compares to our crowdsourced methodology.

It is a infeasible for the collection to contain relevance judgements for all tweets and for every event. Even with a concrete definition of event, such as the one we propose in chapter 3, deciding what constitutes an event is a subjective task, and having mul-

tiple annotators read all 120 million tweets in the collection to annotate all events and all relevant tweets is an impossible task. For that reason, the pooling strategy we employed in chapter 3 is likely the best approach for building a large scale collection for event detection. However, this raises an number of evaluation issues that need to be examined, and if possible, addressed.

Table 4.10: Results obtained through crowdsourcing vs automatically at a minimum event size of 30.

	Automatic	**Crowdsourced**	**Crowdsourced (Scaled)**
Precision	242/1210 (0.200)	159/250 (0.636)	769/1210 (0.636)
Recall	194/506 (0.383)	-	-

Table 4.10 shows the results of our entity-based approach evaluated against the Events 2012 collection using both automated and crowdsourced evaluation methodologies. Of the 250 events evaluated using crowdsourcing, 159 were determined to be true events, while 91 were found to be false events, giving a precision of 0.636. By scaling these values to all events returned by our approach, we can estimate that 769 of the 1210 events are true events. Precision under a crowdsourced evaluation is more than 3 times higher than calculated using the automated approach, indicating as expected, that the Events 2012 collection does not have relevance judgements for all events that occurred during the 28 days it covers.

As we discuss later in this section, there are a number of issues that prevent us from accurately calculating recall for the crowdsourced evaluation, so recall is not given in Table 4.10. The recall value given in Table 4.1 was calculated automatically using existing relevance judgements.

Table 4.11: The distribution of events between categories, measured using both the Collection and Crowdsourcing.

Category	**Automatic**	**Crowdsourced**
Armed Conflicts & Attacks	26.3%	3.4%
Arts, Culture & Entertainment	6.2%	9.4%
Business & Economy	4.6%	1.7%
Disasters & Accidents	6.7%	3.4%
Law, Politics & Scandals	27.8%	21.4%
Miscellaneous	2.1%	11.1%
Science & Technology	2.1%	2.6%
Sports	24.2%	45.3%

Table 4.11 shows the distribution of events between categories, calculated automatically and through crowdsourcing. Both the automatic and crowdsourced evaluation give similar distributions ($r = 0.59$), suggesting that there is at least moderate positive correlation between the distribution of results returned by both methods of evaluation, despite differences in the absolute values.

4.5.1 Evaluation Issues

Calculating recall for the crowdsourced evaluation poses a number of issues. Even though our entity-based approach has discovered new events, leading to a much higher precision score, we cannot be certain how many of these are truly new events or if they are sub-events of events already in the relevance judgements. This is a particular problem for large events that may have many thousands, or in some cases, millions of tweets discussing them. For example, we cannot expect to ever have full relevance judgements for even a fraction of the millions of tweets that were posted about the US Presidential debates in 2012[3].

One possible solution is to use a clustering approach similar to that used in chapter 3, and expand the relevance judgements to include both new events, and new tweets for existing events. However, this would require a crowdsourced evaluation of all candidate events for every run, a prohibitively costly and time intensive task. Due to time and resource constraints, we evaluated only 250 of the 1210 candidate events using crowdsourcing, however even evaluating this smaller set of events was a time consuming task, despite having all the tools and infrastructure already in place. Whilst this does allow us to estimate precision across all 1210 events, it only tells us which of the 250 crowdsourced events are true events, it does not tell us which of the remaining 960 are. Without first removing all false events, we cannot accurately use the clustering technique to merge events and determine how many new events have been detected, meaning that an accurate recall measurement is impossible without a crowdsourced evaluation of all events.

Automatically Determining Relevance

Even for events already in the relevance judgements, it can be difficult automatically determine if a candidate event is relevant. Different clustering approaches and thresholds mean that two systems are unlikely to ever produce an event containing

[3] http://www.nbcnews.com/technology/presidential-debate-sets-twitter-record-6281796

identical tweets. Since we know that the relevance judgements are incomplete in terms of tweets for each event, and that an event detection system is unlikely to correctly identify every tweet for an event, we must allow for inexact matching when determining relevance. Rather than requiring that every tweet matches for a candidate to be determined relevant, a threshold must be set that is both low enough to enable partial matches, but still identifies any true-negatives. Additionally, because automated methods were used to generate the relevance judgements, each with differing levels of granularity, there are a number of events in the judgements where only part of an event has been detected (for example, a single goal in a football match, rather than the football match itself). This means that a high threshold will make it difficult for an event to found relevant if the system being evaluated operates at a different level of granularity from the relevance judgements. For example, the full football match rather than the individual goal, or grouping many small but related events into 'super-event', such as Hurricane Sandy causing damage as it moved across the United States. These systems should not have a lower recall than a system which reports each event individually (although some measure of event granularity would be useful).

Empirically, we found if 5% or 15 tweets (whatever the smallest is) can be matched from a candidate events to an event in the relevance judgements, then they are almost always correctly labelled as relevant. While this may seem like a low threshold, empirically, we found that it produce very few false-positives (i.e. non-relevant events being identified as relevant), whilst allowing for a great deal of flexibility in terms of event granularity. More work is required to develop a more robust evaluation methodology.

4.6 Efficiency and Ensuring Real-Time Processing

One of our aims when developing our entity based approach was to ensure that it was both real-time and efficient. In essence, a real-time algorithm can be seen as having $O(1)$ complexity. While a real-time approach is often efficient, it does not have to be, and many real-time approaches fail to scale as the volume of information is increased, causing effectiveness to drop. Clearly a real-time system cannot be expected to process an infinite volume of data, however an efficient real-time systems should be able to scale reasonably without a significant loss of effectiveness.

Our approach is both real-time and efficient, in that we guarantee a maximum time per tweet, and it is able to scale as the volume of data increases. The real-time aspect is guaranteed by using only the top 10 IDF-weighted terms in each tweet (a restriction

which effects less than 0.02% of tweets) and by retrieving only the N most recent documents per term (in our experiments $N = 500$), we ensure that in the worst case, no more than $10N$ comparisons are made per document, giving $O(1)$ complexity.

The efficiency of our approach comes from a number of aspects. Firstly, as discussed in section 4.2, our approach only uses tweets which contain named entities. This means that the vast majority of noisy and unspecific tweets can be removed before computationally expensive clustering or analysis is performed. This greatly reduces the number of tweets which need to be clustered and the the number of comparisons which need to be made. Although Named Entity Recognition itself can be computationally expensive, this can be scaled across any number of machines before tweets reach the event detection stage. Secondly, the average number of comparisons made per tweet is extremely low. Table 4.12 shows the complexity, worst case, and average number of comparisons for baseline approaches and our entity based approach. Although our worst case performance requires up to 5000 comparisons (this can be lowered by reducing N, the maximum number of documents retrieved per term), the average number of comparisons per tweet is only 72, a tiny fraction of the worst case, and significantly below that of the CS and LSH approaches. This is due to our entity-based clustering approach. By only comparing tweets that contain overlapping named entities, we greatly reduce the candidate set, and thus the number of comparisons that need to be made to find a nearest neighbour. This partitioning has the added benefit that tweets could theoretically be distributed across multiple servers for processing, using the named entities in a tweet to determine which server processes it. Finally, our pipeline architecture allows for each component to work independently and in parallel, allowing it to scale more easily.

Table 4.12: Different complexity aspects of our detection approach and the 2 baselines approaches. The average complexity for LSH was calculated without the use of a variance reduction technique which would push the average higher.

Approach	**Complexity**	**Worst Case**	**Average**
Entity	$O(1)$	5,000	72
LSH	$O(1)$	3,000	210
CS	$O(1)$	1,200	1200

4.7 Conclusion

In this chapter, we proposed a novel, efficient and real-time event detection technique, which uses named entities to partition a stream of incoming tweets and perform clustering. We show how a light-weight burst detection scheme can be used effectively to detect bursty entities, and how semantic links can be formed between bursting entities to describe real-world events. We performed what we believe was the first in-depth analysis of an event detection approach for Twitter, using a large-scale collection of 120 million tweets and over 500 events. We validated our hypothesis using the collection and through crowdsourcing, showing that our approach makes significant improvements over two state-of-the-art baselines and performed in-depth analysis of the results. Finally, we describe a number of issues with automatic evaluation of event detection on Twitter and describe a number of possible solutions.

CHAPTER 5

Scoring Tweets for Newsworthiness

The unpredictable nature of news, combined with the limited length and unstructured nature of Twitter makes it very difficult to predict the newsworthiness of an individual tweet, especially in real-time. Classifying tweets as newsworthy or noise is a common step in credibility scoring, where the aim is to determine if a tweet is newsworthy, and then to decide how credible information contained in the tweet is. However, the newsworthiness of an individual tweets is often ignored in event detection, where clusters of documents or terms are examined, rather than individual tweets.

In this chapter, we examine how newsworthiness can be predicated at a more fine-grained tweet level, and attempt to predict the newsworthiness of individual tweets, which we then use as feature for event detection. The aim of this work is to enable event detection approaches to more easily separate signal (newsworthy tweets) from noise (general discussion and chatter), and to demonstrate that event detection approaches can move away from existing cluster or graph based significance measures, such as the volume of tweets mentioning a specific term or the size of a cluster. We aim to show that event detection approaches can instead rely on latent features that allow for the higher precision while requiring fewer tweets.

For the purpose of this work, we say that a tweet is newsworthy if it discusses a topic that is of interest to the news media. For example, a tweet describing an apartment fire would be considered newsworthy, whereas a tweet describing a user's breakfast would not. Note that the concept of newsworthiness with regards to an individual tweet is very similar to the definition of *significant* in the context of events, defined in chapter 3.1, however applied at a tweet level rather than over an event as a whole. Although still subjective, a topic is either considered significant enough to be an event, or it is not. Newsworthiness, on the other hand, can be measured on a scale. Whilst a tweet asking others to "pray for those involved" in the aforementioned apartment fire is relevant to a newsworthy event because it mentions the apartment fire, it is con-

siderably less newsworthy than a tweet providing information about the fire or the people involved. The aim of this work is to determine whether an individual tweet is newsworthy (its *direction*), and assign a relative score (its *magnitude*).

Given the above, we define 3 key characteristics that we want our Newsworthiness Scorer to satisfy:

- **Generalizability:** as specified in chapter 1, the scoring system should be able to score a tweet about any event or event type, even if it has never encountered a similar event before. It should not, for example, only be able to score tweets about sports events or earthquakes, rather it should be general enough to accurately score tweets about anything that could be reported by the media (i.e., anything *significant*).
- **Real-time:** also specified in chapter 1, the real-time characteristic means that tweets should be scored as soon as received, and there be should be no delay to wait for new information (i.e., no batch processing). This ensures that the scoring is useful for real-time event detection systems.
- **Responsive:** it should be possible to update the model and incorporate new information as it is found. For example, as an event detection system discovers new events or tweets related to ongoing events, it should be able incorporate new information into the scoring model, allowing subsequent tweets to be more accurately scored.

To meet these characteristics, we propose a novel newsworthiness scoring approach which boosts or reduces a tweet's Newsworthiness Score based on the likelihood ratio of a term with regards to term distributions across different tweet qualities. This approach requires no manually annotated training data and instead relies on a small set of heuristics to identify training data in real-time. We can then bolster the models by feeding exceptional tweets, identified either through event detection or using their Newsworthiness Score, back into the models. This allows the scoring model to quickly adapt to new events and information in real-time, giving it a significant advantage over approaches which use only manually labelled training data.

The intuition here is that by identifying high and low quality tweets, we can compare term usage statistics to the corpus as a whole, giving us an insight into which terms are newsworthy, or which add noise. By identifying terms (or groups of terms) that appear more commonly in newsworthy tweets when compared to the background corpus, and measuring their significance using likelihood ratios, we can assign a Newsworthiness Scores to a tweet by combining individual term scores.

In this chapter, we aim to answer the following research questions:

- Can heuristics be used to automatically train newsworthiness scoring models?
- Can an automatic approach identify the newsworthiness and magnitude of a tweet?
- Can newsworthiness be used as a feature to improve event detection approaches?

In section 5.2 we describe a set of heuristics for automatically classifying content into high and low quality sets, which can then be used to train models for newsworthiness scoring. We then use these models to identify and score newsworthy content, evaluating the effectiveness of our proposed approach in section 5.4. Finally, in section 5.5, we demonstrate how our newsworthiness scoring approach can be used to improve upon a simple cluster-based event detection approach and outperform current state of the art event detection approaches.

5.1 Background

The automatic classification of story newsworthiness is not new, and is a common pre-processing step in credibility assessment and summarization [Noyunsan et al. 2017; Madhawa and Atukorale 2015].

Noyunsan et al. [2016] used cosine similarity to filter out non-newsworthy posts on Facebook by comparing new posts to known non-newsworthy posts. They created a groundtruth of 1,005 non-newsworthy Facebook posts, annotated and collected via a browser extension installed by volunteers. Their approach compares each new post to every post in the groundtruth, and classifies posts as non-newsworthy (noise) if its similarity to any document in the groundtruth is above a threshold. They later examined how the removal of non-newsworthy posts affects credibility assessment and found that by removing non-newsworthy posts they could increase the credibility assessment effectiveness [Noyunsan et al. 2017].

Madhawa and Atukorale [2015] examined the performance of difference classifiers for labelling tweets as newsworthy or noise as the first step in a pipeline for the summarization of events on Twitter. However, Madhawa and Atukorale [2015] define newsworthy and noise as 'objectivity' and 'subjectivity' respectively. Although this distinction is true in many cases, and provides a useful distinction for summarization, it does not apply in all cases. Subjective comments by noteworthy individuals, such as politicians or public figures, are often a newsworthy event. For this reason, we argue that simple objectivity and subjectivity labelling does not capture the true newswor-

thiness of a post, and instead a more robust newsworthiness *score* is required.

5.2 Heuristic Labelling and Quality Classification

We propose an automatic labelling approach using a set of heuristics to label high quality (newsworthy) and low quality (noisy) content. Content is labelled in streaming fashion to simulate real-world conditions, and is used as training data for our newsworthiness scoring model. Note that our labelling approach is specifically designed not to label the majority of content. Instead, we want to identify exceptional content which we can compare to the 'random' background corpus, allowing the model to learn specific newsworthy and noisy features.

The use of heuristics has a number of advantages over the use of existing datasets and requires only minimal effort in comparison to creating a labelled dataset specifically for this task. Many existing datasets that could be used for newsworthiness prediction are extremely small, or focus on a single large event or topic [Kang et al. 2012; Madhawa and Atukorale 2015], making them useful only for training a newsworthiness classifier for those specific events, and unfeasible for training a broad and event-agnostic newsworthiness classifier.

Even large datasets with relevance judgements, such as the Events 2012 corpus from chapter 3, covering 506 events with more than 100,000 relevance judgements, are not suitable for supervised training. Datasets of this size, although useful for evaluation, make no claims of completeness. Despite having relevance judgements for a large number of events and tweets, the Events 2012 corpus covers only covers a tiny fraction of the many thousands of newsworthy events that take place every day, so training specifically on these events will likely result in over-fitting and decreased effectiveness on other datasets.

The use of heuristic labelling, rather than manual labelling, allows for training data to be label in real-time and fed into the scoring approaches that follow. This allows our approach to learn new features in real-time, as described in section 5.3.

Madhawa and Atukorale [2015] used heuristic labelling to generate training data for newsworthiness classification. They developed a classifier capable of classifying documents as either objective (high quality, newsworthy), or subjective (low quality, noise) as a filtering step before summarization. However, their approach used only a small, manually curated lists of accounts as newsworthy sources, and labelled any tweet containing an emoticon or emoji as noise. Instead, we use a broader set of heurist-

ics that place fewer restrictions on what constitutes newsworthy and noisy sources, allowing for a broader range of sources and more training data.

As the inclusion of 'conversational' or non-newsworthy posts has been shown to have a negative effect on credibility assessments [Noyunsan et al. 2017; Sikdar et al. 2013b], many credibility assessment approaches use features that attempt to capture the newsworthiness of a post before assessment. We base many of the heuristics that follow on those shown to be effective for credibility assessment [Sikdar et al. 2013a; Kang et al. 2012; Castillo et al. 2011; Madhawa and Atukorale 2015].

Although these heuristics form an important part of this work, as we demonstrate in section 5.4, the exact choice of features and weights is almost inconsequential, as we use a very simple score cut-off to determine quality. As long as the heuristics can be used to provide a reasonable level of accuracy, the performance of the Newsworthiness Scorer is only mildly affected by changes to the heuristics or their weights.

5.2.1 Features

We define a number of weighted heuristics designed to identify exceptionally high or low quality content. Weights are multiplicative, and we take the product of the weights from each feature as the overall Quality Score, Q_d.

User Description: W_{desc}

We manually created a list of keywords and phrases commonly used by news broadcasters, journalists and financial traders in their User Description on Twitter. The full list of these terms and their weights can be found in Table 5.1.

Table 5.1: Terms and weights assigned to each term for scoring a user's profile description.

Type	Weight	Terms
News & Journalism	2.0	news, report, journal, write, editor, analyst, analysis, media, updates, stories
Finance & Trading	2.0	trader, investor, forex, stock, finance, market
Spam	0.1	ebay, review, shopping, deal, sales, marketing, promot, discount, products, store, diet, weight, porn, follow back, followback

Positive terms were identified manually based on common terms used by news organizations, journalists, and financial traders on Twitter. The list of spam terms was also

created manually based on common types of low quality marketing spam that is often found on Twitter. The 'follow back' terms are commonly associated with users who artificially inflate their follower count by 'following back' anyone who follows them. We penalise them here to correct for any boost given by the Number of Followers feature described later in this section.

Rather than match whole words, we match on a prefix basis. For example, 'report' matches *report*, as well as *report*er and *report*s. Note that this is similar, but not identical to, matching on stemmed terms. This helps to keep the list of terms short whilst giving high coverage. The overall feature score is a product of weights for any matching terms.

Account Age: W_{age}

Young accounts are generally viewed as less credible than accounts that have been active for longer periods of time [Sikdar et al. 2013a], so we give accounts created within the last 90 days a lower weight.

Table 5.2: Follower ranges and weights assigned to accounts who have followers between the range defined.

Account Age (days)	Weight
< 1 day	0.05
< 30 days	0.10
< 90 days	0.25
90+ days	1.00

Number of Followers: $W_{followers}$

It is often the case that something becomes news not because of what was said, but who said it. A head of state or public figure expressing sympathy for victims of an accident is considerably more newsworthy than the average person doing the same.

The number of followers a users has can infer how influential or newsworthy the user is, and thus how newsworthy the user's tweets are likely to be, and is a commonly used feature used for automatic credibility assessment [Kang et al. 2012; Sikdar et al. 2013a; Gün and Karagöz 2014; Madhawa and Atukorale 2015]. Given this, we assign higher weights to users with more followers, and a lower weight to users with very few followers, as shown in Table 5.3.

The vast majority of tweets (83.81%) are posted by users who have between 50 and 4,999 followers, and are unaffected by this feature. The aim is to affect only the extremes: users with very few, or very many followers.

Table 5.3: Follower ranges and weights assigned to accounts who have followers between the range defined.

Number of Followers	Weight
0 - 49	0.5
50 - 4,999	1.0
5,000 - 9,999	1.5
10,000 - 99,999	2.0
100,000 - 999,999	2.5
1,000,000+	3.0

User Verified Status: $W_{verified}$

Politicians, organizations, celebrities, journalists and other public figures can request that Twitter 'verify' their identity by marking their account as verified and displaying a blue badge near to their name or display picture [1]. Although the exact requirements for verification are undocumented, verification is often seen as a sign of authenticity and significance.

At the time of writing, approximately 290,000 accounts have been verified by Twitter, a full list of which can be obtained by examining the list of accounts followed by Twitter's `@verified`[2] account. A survey of Verified accounts in 2015 found that approximately 41% of account are related to news, journalism, politics or government[3]. This supports our hypothesis that verified accounts are a good source of high quality, newsworthy content, so we give Verified users a weight of 1.5. Unverified users as unaffected by this feature (i.e. given weight of 1.0).

Posts Per Day: W_{ppd}

Quality and quantity often have an inverse correlation, especially on social media. Accounts which produce an extremely high volume of posts are often automated ac-

[1] `https://help.twitter.com/en/managing-your-account/about-twitter-verified-accounts`

[2] `https://twitter.com/verified/following`

[3] `https://medium.com/three-pipe-vc/who-are-twitter-s-verified-users-af976fc1b032`

counts repeating content from other sources with the aim of acquiring followers, advertising a product or service, and more recently, for the purpose of propaganda and misinformation [Forelle et al. 2015; Howard and Kollanyi 2016].

Accounts that post more than 50 times per day are often considered to be heavily automated [Howard and Kollanyi 2016]. For this reason, we penalize any account that posts more than 50 times per day on average (weight of 0.5), and apply a more severe penalty for accounts which more than 100 times per day on average (weight of 0.25).

We note, however, that many heavily automated accounts are in fact prominent news and media organizations. To prevent these legitimate accounts from being penalized, we do no apply any penalty to Verified accounts.

Has Default Profile Image: W_{image}

Twitter users who do not provide a custom profile image (often nicknamed "eggs" due to Twitter's historic use of a egg as the default profile image) are generally considered less trustworthy and credible [Castillo et al. 2011; Sikdar et al. 2013a; Gün and Karagöz 2014] than users who have taken the time to provide a custom image. Twitter themselves noted that users who create 'throwaway' accounts, often for the purpose of spamming or to post abuse, tend not to personalize their accounts. The association between the default egg image and low quality accounts has been noted in published work previously [Sikdar et al. 2013a], and the public association was one of the key reasons noted by Twitter for changes to their default profile image in March 2017[4].

We assign user accounts that have not specified a custom profile image a weight of 0.5.

5.2.2 Labelling

As described earlier, document d is assigned an overall Quality Score, Q_d, taken as the product of scores from each feature:

$$Q_d = W_{desc} \times W_{age} \times W_{followers} \times W_{verified} \times W_{ppd} \times W_{image}$$

For example, a tweet d with weights 2.0 for W_{desc}, 1.5 for $W_{followers}$, and 1.0 for the all other features would have an overall Quality Score, $Q_d = 2.0 \times 1.0 \times 1.5 \times 1.0 \times 1.0 \times 1.0 = 3.0$.

[4] https://blog.twitter.com/en_us/topics/product/2017/rethinking-our-default-profile-photo.html

Cut-off values are used to determine quality labels from Q_s. We examine how various cut-off values affect performance in section 5.4, however unless otherwise stated, we label tweets with $Q_d \geq 4.0$ as High Quality, and $Q_d \leq 0.25$ as Low Quality. Documents between this range are unlabelled.

5.3 Newsworthiness Scoring

As documents are processed in order, they are added to a background model containing all documents. In addition, documents labelled either High or Low Quality are also added to corresponding quality specific models: documents labelled High Quality are added to the HQ model, and documents labelled Low Quality are added to the LQ model.

The background model serves as a 'random' distribution that we can compare against to identify terms that are more likely to appear in newsworthy content than at random, and conversely, terms that are more likely to appear in noise than at random. This concept is somewhat similar to the intuition behind the Divergence From Randomness (DFR) Framework and its associated models [Amati and Van Rijsbergen 2002]: where the more a term's frequency within the HQ or LQ models differs from the term frequency of the collection as a whole, the more weight a term carries.

5.3.1 Term-Based Likelihood Ratio Scoring

The likelihood ratio for term t, $R(t)$, gives us an indication of the relative importance of the term in the particular quality model when compared to the 'random' background model. A likelihood ratio greater than 1.0 means that the term is more common in the model than random, whereas a ratio less than 1.0 means that the term is less common in the model than random. We believe that this can be used to identify terms that indicate newsworthiness (or lack thereof).

The likelihood ratio for a term is calculated for both the High Quality ($R_{HQ}(t)$) and Low Quality ($R_{LQ}(t)$) models as:

$$R_{HQ}(t) = \frac{P(t|HQ)}{P(t|BG)} = \frac{\frac{tf_{t,HQ}}{F_{HQ}}}{\frac{tf_{t,BG}}{F_{BG}}} = \frac{tf_{t,HQ} \times F_{BG}}{tf_{t,BG} \times F_{HQ}}$$

$$R_{LQ}(t) = \frac{P(t|LQ)}{P(t|BG)} = \frac{\frac{tf_{t,LQ}}{F_{LQ}}}{\frac{tf_{t,BG}}{F_{BG}}} = \frac{tf_{t,LQ} \times F_{BG}}{tf_{t,BG} \times F_{LQ}}$$

where HQ, LQ and BG correspond to the High Quality, Low Quality and Background models respectively. tf_t is the raw frequency of term t in the given model, and F is the raw frequency of all terms in the model (i.e. $F = \sum f_t$). Since documents are added to each model as they are labelled, but before scoring, we can guarantee that division by 0 will never happen.

For each term t, we define an score S_t for each of the High and Low quality models. The aim is to use the HQ likelihood ratio to boost scores for documents containing terms associated with newsworthy content, and the LQ likelihood ratio to dampen scores for documents containing terms associated with noise. Note that we do not assign scores for every term. Instead, we only use the likelihood ratio for terms with ratios of at least 2.0 in one of the two models, and instead assign a score of 0 where the term has a ratio of less than 2.0 for both the HQ and LQ models. This prevents terms which have no clear association with either high or low quality content from affecting the overall score of a document.

$$S_{HQ}(t) = \begin{cases} R_{HQ}(t), & \text{if } R_{HQ}(t) \text{ or } R_{LQ}(t) \geq 2.0 \\ 0, & \text{otherwise} \end{cases}$$

$$S_{LQ}(t) = \begin{cases} R_{LQ}(t), & \text{if } R_{HQ}(t) \text{ or } R_{LQ}(t) \geq 2.0 \\ 0, & \text{otherwise} \end{cases}$$

Combining Terms Scores

We calculate the overall Newsworthiness Score N_d of a document d as follows:

$$N_d = \log_2 \left(\frac{\frac{1+\sum_{t \in d} S_{HQ}(t)}{D}}{\frac{1+\sum_{t \in d} S_{LQ}(t)}{D}} \right) = \log_2 \left(\frac{1 + \sum_{t \in d} S_{HQ}(t)}{1 + \sum_{t \in d} S_{LQ}(t)} \right)$$

where t refers to a term in document d which contains D number of terms. As D is constant, it can be ignored, simplifying the calculation.

We calculate the average term score $S(t)$ (as described earlier) against both the high and low quality models. By taking the ratio of these scores, we can estimate how newsworthy the terms used in the document are overall. Documents where the average high quality term score is greater than that of low quality terms (i.e. $\sum_{t \in d} S_{HQ}(t) > \sum_{t \in d} S_{LQ}(t)$) are considered newsworthy, whilst documents where the average low

quality term score is greater are considered noise.

We add 1 to both the numerator and denominator which has a number of benefits. Firstly, it smooths the value, dampening the effect of high or low quality terms. This has a larger effect when the number of terms is small (i.e. when there is less evidence), and a smaller effect when there are more terms (i.e. more evidence). Secondly, it prevents division by 0 and $log(0)$, both of which are undefined. In cases where there was no evidence for or against newsworthiness (i.e. no high or low quality terms are found), this ensures that the score N_d is $log(1) = 0$.

Finally, we take log_2 of the score ratio. This ensures that scores are smoothed, dampening the effect of outliers. Using log ensures that scores are centred around 0: tweets with a ratio greater than 1 are given a positive score, whilst tweets with a ratio less than 1 are given a negative score. Although it may seem counter intuitive for a tweet to have a negative Newsworthiness Score, a negative score helps to better represent that the tweet is both non-newsworthy, and is likely to be of very low quality or spam. In other-words, it highlights that the tweet is noise.

5.4 Evaluation

Since there is no existing collection specifically for the evaluation of newsworthiness, and our overall aim is to develop a score that can be used to improve real-time event detect techniques, we evaluate our newsworthiness scoring approach on the Events 2012 corpus created and described in chapter 3, a corpus created specifically for the evaluation of event detection approaches.

First we examine the different heuristic features defined in section 5.2 in the context of the Events 2012 corpus. We then test a number of different threshold values for labelling tweets as high and low quality, and examine how Newsworthiness Scores are distributed. We look at how different event types and categories behave, and examine how different term representations (unigrams, bigrams) affect performance.

We process the Events 2012 corpus in a sequential manner, simulating a real-time stream of tweets. We filter out retweets and any non-English tweets. URLs are removed, and each tweet is tokenized. Unless otherwise noted, a unigram representation is used, we perform no stemming or stop word removal, and all tokens are normalized to lowercase.

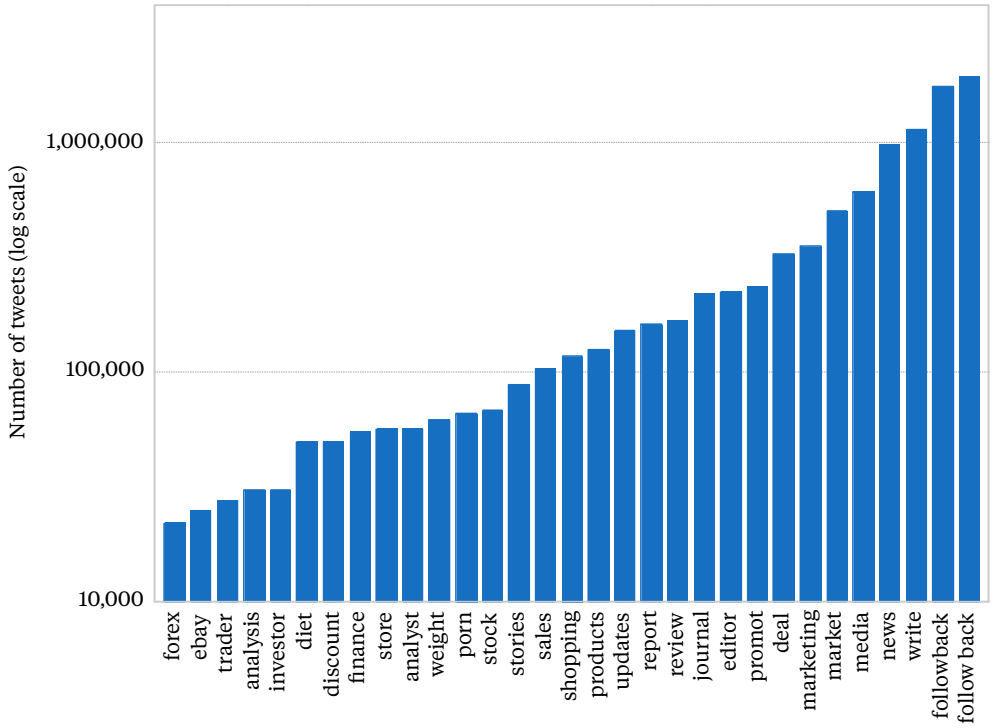

Figure 5.1: The total number of tweets posted by users with any of the terms listed in Table 5.1, sorted by frequency.

5.4.1 Heuristics

In section 5.2, we defined a number of heuristic features used to calculate a basic Quality Score that we then use to label tweets as either high or low quality. We examine how these heuristics behave against tweets from the Events 2012 corpus, in terms of the percentage of tweets affected by each heuristic. Although our heuristics are all user-based, rather than content-based, we report the number of tweets, rather than the number of unique users, as the volume of tweets affects the performance of our approach, not the number of unique users.

User Description

Figure 5.1 shows the number of tweets posted by a user whose User Description contains one of our heuristic terms, sorted by raw frequency. Note that the most common phrases "follow back/followback" refer to a common type of spam described in section 5.2, where a user promotes the fact that they will follow any user who follows them. This is often used to artificially grow followers and increase the accounts perceived significance. Since we use high numbers of followers as a positive weight, it is important that we are able to detect these accounts appropriately.

Terms relate to news and journalism appear high in the list: 'media', 'news' and 'write' are the most common non-spam related terms, with other news and journalism related terms ('report', 'journal', 'editor') appearing slightly lower in terms of raw frequency. These are extremely strong indicators that the user is either a news broadcaster or a journalist.

Finance related terms appear less frequently, with the exception of 'market'. However, the bulk of this is due to the term 'market' also matching 'marketing', which is the next most common term. Although 'market' has a weight of 2.0, tweets by users whose description contains the phrase 'marketing' (weight 0.1) will still be given an overall weight of $2.0 \times 0.1 = 0.2$, ensuring that the overall Quality Score is kept low. Despite the overall low frequency of financially related terms, we feel that their inclusion is important to capture discussion of financial news which may be discussed less broadly than other types of news.

Followers

Table 5.4 shows the number of tweets from users with particular follower ranges, and percentage of total tweets being posted by each group. As mentioned in section 5.2, the vast majority (83.81%) of tweets are posted by users with between 50 and 4,999 followers. This means that for the majority of tweets, this feature has no effect on the Quality Score since this range is given a weight of 1.0. However, for tweets posted by users with fewer than 50 followers, which make up 13.78% of tweets in the collection, this feature has a negative effect on the Quality Score. Very few tweets come from users with more than 1 million followers, a total of only 9,457 tweets, or 0.01% of tweets in the collection, however this relatively small number of tweets will receive a significant boost in their Quality Score ($W_{followers}$ = 3.0).

Table 5.4: Follower ranges and and the number of tweets posted by users (excluding retweets) within the given range of followers.

Number of Followers	Tweets	%
0 - 49	10,358,082	13.78%
50 - 4,999	63,006,739	83.81%
5,000 - 9,999	806,565	1.07%
10,000 - 99,999	905,533	1.20%
100,000 - 999,999	90,620	0.12%
1,000,000+	9,457	0.01%

Posts Per Day

Table 5.5 shows the number of tweets posted by users who post the specified numbers of tweets per day on average since creating their account. We can see that although the vast majority of tweets are posted by users who tweet less than 50 times per day on average, 12.4% of tweets come from users who post more than 50 timer per day on average, meaning that these tweets will receive a Quality Score penalty under our heuristic scoring approach.

Table 5.5: The number of tweets in the collection (excluding retweets) from users who post various volumes of tweets per day, on average.

Average Posts Per Day	Tweets	%
< 50	65,856,339	87.60%
50 - 100	6,334,083	8.43%
> 100	2,986,574	3.97%

Account Age

Table 5.6 shows the distribution of tweets posted by users with accounts of varying age. Over 90% of tweets are posted from accounts older than 90 days, however a not-insignificant number of posts come from accounts younger than 90 days. A surprisingly large number of tweets, over 1%, come from accounts created in the past 24 hours, and 9.23% from accounts created in the past 90 days.

Table 5.6: The number of tweets in the collection (excluding retweets) from users who post various volumes of tweets per day, on average.

Account Age	Tweets	%
< 1 day	786,989	1.05%
1-29 days	2,286,548	3.04%
30-90 days	3,863,935	5.14%
> 90 days	68,239,524	90.77%

Verified Users

The collection contains 110,768 tweets posted by Verified users, accounting for 0.15% of the total tweet volume (excluding retweets).

Default profile image

The collection contains 1,574,774 tweets posted by users with the default profile image, accounting for 2.09% of the total tweet volume (excluding retweets).

5.4.2 Quality Scores

Figure 5.2 shows the percentage of tweets assigned a Quality Scores $Q_d < 1.0$. A total of 24,713,071 (32.87%) tweets were given a 'low' Quality Score of less than 1.0. A large number of tweets have scores of precisely 0.5 or 0.25 (10,270,675 and 3,903,268 respectively) due to the particular weights assigned to features such as Account Age and Average Tweets Per Day. The average low Quality Score is 0.29, with a median of 0.25.

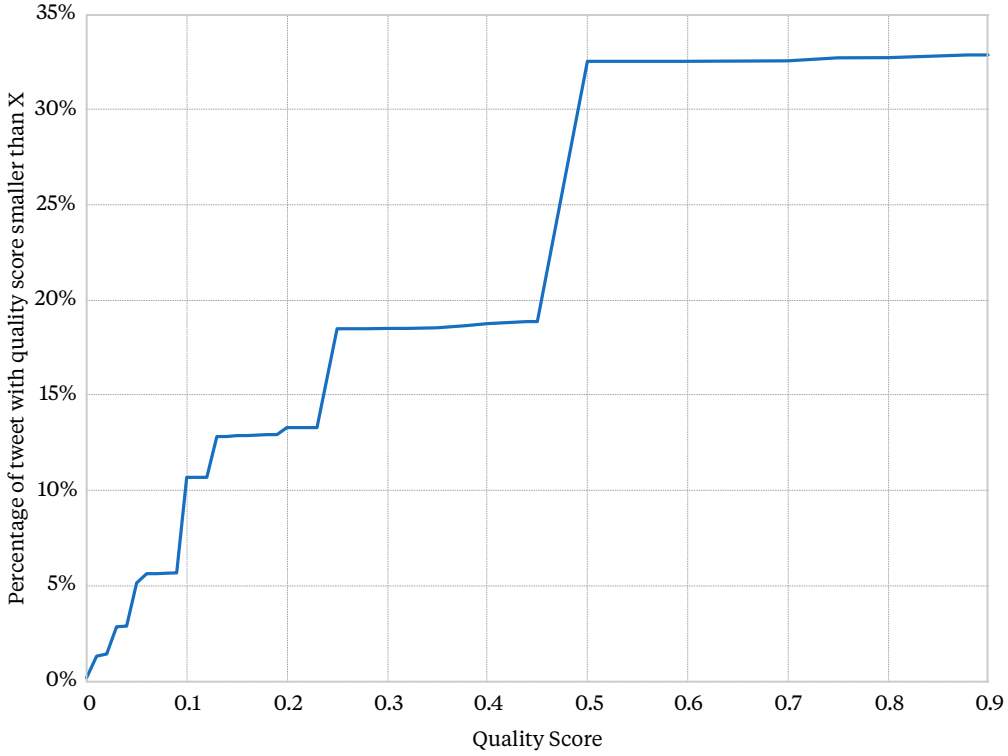

Figure 5.2: Cumulative percentages of tweets with Quality Score Q_d lower than the value on the x-axis, up to a maximum of 1.0

Similarly, Figure 5.3 shows the percentage of tweets assigned 'high' Quality Scores (i.e. $Q_d > 1.0$). Note that only 3,230,830 (4.30%) of tweets are assigned scores greater than 1.0, significantly fewer than assigned a score of less than 1.0. The average high Quality Score is 2.46, however this is warped by a small number of outliers with extremely high Quality Scores (19 tweets have the maximum score of 224). The median high Quality Score is 2.0.

A total of 47,233,095 (62.83%) tweets are assigned a Quality Score of exactly 1.0, meaning that they were either assigned weights of 1.0 for all heuristic quality features, or assigned scores that cancelled. The median score across all tweets is 1.0.

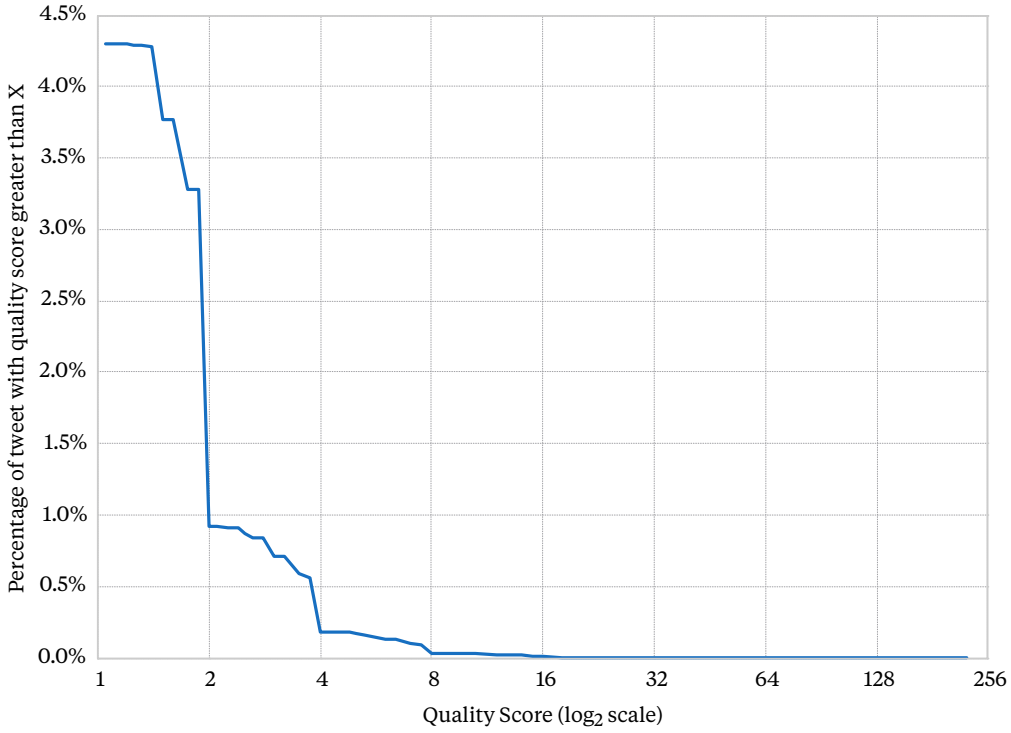

Figure 5.3: Cumulative percentage of tweets with Quality Scores Q_d higher than the value on the x-axis. The x-axis uses a log_2 scale as the distribution is heavily skewed towards 1.

5.4.3 Quality Thresholds

We examine how different thresholds affect the overall performance of our newsworthiness predictions. Since we lack a groundtruth with appropriate Newsworthiness Scores, we instead examine the differences between scores and newsworthiness classifications for tweets known to be relevant to events in the Events 2012 corpus, and those that were not evaluated. It is important to note that we say 'not evaluated', rather than non-relevant, as we cannot be certain that tweets that were not evaluated are not newsworthy, since only a fraction of the 120 million tweets in the Events 2012 corpus were evaluated to create relevance judgements. It is likely that there are a significant number of unevaluated, but newsworthy tweets not included in the relevance judgements. This means that making absolute statements is impossible. Instead we must assume that the percentage of unevaluated but newsworthy tweets is low enough not to mask any differences in classification rates or scores.

Quality Labels

Table 5.7 shows the percentage of tweets labelled as High Quality or Low Quality. We divide tweets into two sets: those known to be relevant to an event from the Events 2012 corpus, and all other tweets (denoted 'Event' and 'Other'). We also include the ratio of Event / Other tweets so that the relative percentages can be more easily compared as Quality Score thresholds are varied. The threshold values represent the minimum or maximum scores for a tweet to be labelled as High Quality or Low Quality respectively. For example, a tweet with a Quality Score of 2.00 would be labelled High Quality at threshold values of 2.00 and below, but not at threshold values of 4.00 or 8.00. Similarly, a tweet with a Quality Score of 0.67 would be classified as Low Quality only if the threshold value was set to 0.67 or above.

Table 5.7: Percentages of Event and Other tweets given High Quality and Low Quality labels at a range of Quality Score thresholds.

	High Quality		
Quality Score	**Event**	**Other**	**Ratio**
≥ 1.25	12.346%	4.286%	2.881
≥ 1.50	12.313%	4.275%	2.880
≥ 2.00	11.505%	3.755%	3.064
≥ 4.00	3.345%	0.677%	4.944
≥ 8.00	0.772%	0.116%	6.626
≥ 16.00	0.149%	0.017%	8.546
	Low Quality		
Quality Score	**Event**	**Other**	**Ratio**
≤ 0.25	15.358%	18.497%	0.830
≤ 0.50	31.713%	32.536%	0.975
≤ 0.75	32.041%	32.713%	0.979

Ratios greater than 1.0 for High Quality labels suggests that Event tweets are more likely to be labelled as High Quality than a tweet selected at random. Conversely, a ratio of less than 1.0 for Low Quality labels means that Event tweets are less likely to be labelled as Low Quality than a tweet selected at random.

A clear difference can be seen between Event and Other tweets. A much higher percentage of Event tweets are labelled High Quality at each threshold compared to Other tweets, with a general trend suggesting the ratio increases with threshold for High Quality labels: as the High Quality score threshold is increased, the ratio increases from 2.881 at 1.25 to 8.546 at 16.00. A similar, but inverse and weaker trend can be

seen as the threshold is decreased for Low Quality labels.

This is a good indication that the selected heuristics are reasonable choices, and perform better than random at labelling content, as a random baseline would show ratios close to 1 for both Low and High Quality labels.

5.4.4 Effect of Quality Score Thresholds on Newsworthiness Classification

Table 5.8 gives the percentage of tweets classified as Newsworthy or Noise (i.e. Newsworthiness Scores that are greater or less than 0, respectively), across a range of Quality Score threshold combinations. At all Quality Score thresholds tested, a higher percentage of Event tweets are classified as Newsworthy and a smaller percentage are classified as Noise, compared with Other tweets. Compared to a random baseline, which would give similar classification rates for both Event and Other tweets, our newsworthiness classification appears to perform well. Event tweets are classified as Newsworthy at a ratio of approximately 4 to 1 when compared to Other tweets, and classified as Noise only one third as often as Other tweets. The deviation from a ratio of 1 suggests that our Newsworthiness Scoring approach is effective at separating newsworthy content from noise.

Varying the LQ threshold appears to have only a small overall effect on the classification rates between 0.75 and 0.5. However, once the threshold is dropped to 0.25, a more substantial change occurs. This can likely be explained by the volume of tweets with scores below these thresholds, which remains around 32% between 0.75 and 0.5, but drops to 18% for 0.25, as can be seen in Table 5.7.

Table 5.8: Percentages of tweets classified as either Newsworthy or Noise for Event and Other tweets, across a range of Quality Score threshold values.

Thresholds		**Newsworthy**		**Noise**	
HQ	**LQ**	**Event**	**Other**	**Event**	**Other**
1.25	**0.75**	49.759%	11.146%	6.649%	19.138%
1.25	**0.50**	50.334%	11.287%	6.696%	19.218%
1.25	**0.25**	61.537%	13.694%	5.736%	19.678%
1.50	**0.25**	61.564%	13.713%	5.738%	19.682%
2.00	**0.25**	64.122%	14.370%	6.601%	22.218%
4.00	**0.25**	76.504%	18.830%	14.567%	50.133%
8.00	**0.25**	72.748%	17.137%	18.834%	66.536%
16.00	**0.25**	65.307%	17.232%	22.650%	67.193%

The percentage of tweets classified as newsworthy reaches a maximum when we use a HQ threshold of 4.00. Referring back to Table 5.7, we can see that only 3.345% of Event tweets have a Quality Score that meets or exceeds this threshold and would be used by the HQ model. Despite this, 76.504% of Event tweets are classified as newsworthy, but only 18.830% of Other tweets are, giving a ratio of 4.063. As the HQ threshold increases, the average tweet quality also increases, but decreases the volume of tweets added to the model. This has the effect of reducing noise, and seems to benefit our newsworthiness scoring approach by increasing the proportional probability of newsworthy terms. However, increasing the HQ threshold offers improvements only to a point, as the percentage of Event tweets given positive Newsworthiness Scores decreases as we increase the HQ threshold above 4.0. The is likely due to the volume of tweets added to the HQ model drops sharply to less than 1% of Event tweets above a Quality Score of 4.0, causing the model to miss important and useful information.

As overall classification ratios seem to reach a maximum around 4.0 and 0.25 for HQ and LQ thresholds, we use these thresholds for all experiments that follow.

5.4.5 Newsworthiness Scores

Table 5.9 shows the percentage of tweets classified as Newsworthy or Noise across a range of Newsworthiness Scores. For Newsworthy tweets, we can see that as the minimum Newsworthiness Score increases towards 4.0, the ratio of Event to Other tweets increases from 4.063 to 10.207; whilst 11.707% of Event tweets have a Newsworthiness Score of 4.0 or greater, only 1.147% of Other tweets do.

Table 5.9: Percentages of tweets classified as either Newsworthy or Noise for Event and Other tweets, across a range of Newsworthiness Score threshold values.

Newsworthy				Noise			
Score	**Event**	**Other**	**Ratio**	**Score**	**Event**	**Other**	**Ratio**
> **0.0**	76.504%	18.830%	4.063	< **0.0**	14.567%	50.133%	0.291
≥ **1.0**	71.109%	16.223%	4.383	≤ **-1.0**	10.262%	47.076%	0.218
≥ **2.0**	59.848%	11.117%	5.383	≤ **-2.0**	6.964%	34.800%	0.200
≥ **3.0**	40.301%	4.701%	8.573	≤ **-3.0**	2.752%	16.797%	0.164
≥ **4.0**	11.707%	1.147%	10.207	≤ **-4.0**	0.575%	5.240%	0.110
≥ **5.0**	1.342%	0.221%	6.078	≤ **-5.0**	0.111%	1.562%	0.071
≥ **6.0**	0.183%	0.046%	3.969	≤ **-6.0**	0.024%	0.519%	0.047
≥ **7.0**	0.030%	0.011%	2.858	≤ **-7.0**	0.014%	0.245%	0.058

Similarly, for Noise tweets, the ratio of Event to Other tweets decreases as we decrease the maximum Newsworthiness Score. The smaller ratio in this case means that Event tweets are less likely to be classified as Noise than Other tweets. We stop at -7.0 as no Event tweet had Newsworthiness Score of -8.0 or lower.

These results are encouraging, and follow a reasonable score distribution. The increasing and decreasing ratios for Newsworthy and Noise tweets respectively suggests that Newsworthiness Scores do incorporate some notion of newsworthiness magnitude, something we examine more closely in sections 5.4.6 and 5.4.7.

5.4.6 Event Categories

Table 5.10 shows the distribution of tweets labelled Newsworthy, Noise, or Unclassified (i.e., Newsworthiness Scores greater than, less than, or exactly 0 respectively) across the different event categories defined by the Events 2012 corpus. Category scores vary considerably around the mean values of 76.504% for Newsworthy and 18.497% for Noise (as shown in Table 5.8). Table 5.11 shows the average overall Newsworthiness Score across all Event tweets for each category, as well as independently calculated averages for tweets classified as Newsworthy or Noise.

Over 97% of tweets discussing Armed Conflicts and Attacks events are classified as Newsworthy, whereas only 42.105% of Arts, Culture & Entertainment tweets are. This disparity can also been seen in the average Newsworthiness Scores for both categories: Armed Conflicts & Attacks has an average Newsworthiness Score of 3.317, whilst Arts, Culture & Entertainment has an average of only 0.427, although this difference is reduced when looking at averages for tweet classified as Newsworthy only (3.428

Table 5.10: The raw counts and percentages per category of tweets classified as Newsworthy or Noise.

Category	Newsworthy (%)	Noise (%)	Unclassified (%)
Armed Conflicts & Attacks	8425 (97.613)	120 (1.390)	86 (0.996)
Arts, Culture & Entertainment	3477 (42.105)	2609 (31.594)	2172 (26.302)
Business & Economy	3589 (90.746)	127 (3.211)	239 (6.043)
Disasters & Accidents	4196 (85.146)	453 (9.192)	279 (5.662)
Law, Politics & Scandals	33197 (89.620)	3031 (8.183)	814 (2.198)
Miscellaneous	2004 (53.698)	856 (22.937)	872 (23.365)
Science & Technology	1958 (82.896)	133 (5.631)	271 (11.473)
Sports	19077 (62.892)	7127 (23.496)	4129 (13.612)

Table 5.11: Average Newsworthiness Scores for each event category, calculated for all tweets, only tweets classified as Newsworthy, and only tweets classified as Noise.

	Average Newsworthiness Scores		
Category	**All Tweets**	**Newsworthy**	**Noise**
Armed Conflicts & Attacks	3.317	3.428	-2.044
Arts, Culture & Entertainment	0.427	2.806	-2.388
Business & Economy	3.260	3.659	-1.880
Disasters & Accidents	2.329	2.913	-1.645
Law, Politics & Scandals	2.620	3.040	-1.274
Miscellaneous	0.803	2.249	-1.764
Science & Technology	2.613	3.294	-2.089
Sports	1.044	2.410	-2.007

and 2.806, respectively). This difference seems intuitive: armed conflicts and attacks, such as bombings, school shootings and assassination attempts are extremely newsworthy events that are likely to make headlines worldwide. On the other hand, entertainment events, such as the launch of new television shows, or award ceremonies such as the Black Entertainment Awards, are less likely to make worldwide headlines but will still generate a high volume of subjective and low quality chatter.

Other categories behave similarly to Armed Conflicts & Attacks, such as Business & Economy, where events are generally headline news, but generate a relatively low volume of discussion. Sports and Miscellaneous events, such as Felix Baumgartner's record-breaking jump from the edge of space or the start of Daylight savings time in the United States, show similar behaviour to Arts, Culture & Entertainment. These types of events generate a high volume of low quality chatter, resulting in a lower percentage of tweets classified as Newsworthy and lower average Newsworthiness Scores.

5.4.7 Example Event: Lenovo overtakes HP

Table 5.12 gives a sample of tweets from Event #81 of the Events 2012 corpus, sorted by Newsworthiness Score from highest to lowest. The event describes how Lenovo, a Chinese computer manufacturer, has taken the top spot from HP in terms of number of PCs sold worldwide. The judgements for this event identify 41 tweets thought to be relevant to the event. Whilst the 36 of the 41 tweets are relevant, our newsworthiness scoring approach correctly identifies a number of spam and non-relevant tweets and

Table 5.12: A sample of tweets and their Newsworthiness Scores from Event #81 of the Events 2012 corpus, sorted by Newsworthiness Score from highest to lowest.

Score	Tweet
4.102	H-P, Lenovo jockey for No. 1 in PCs: reports: SAN FRANCISCO (MarketWatch) Hewlett-Packard and Lenovo were jockey...
3.411	Lenovo knocks HP off top of global PC market: Gartner: SAN FRANCISCO (Reuters) - China's Lenovo Group...
2.177	HP, Lenovo battle for top spot in PC market - Computerworld
2.070	HP, Lenovo battle for top spot in PC market - Computerworld
1.964	Lenovo Overtakes HP as World's Top PC Maker in Q3
1.542	Lenovo passes HP to be top PC maker: The Chinese group has a 15.7% share of worldwide shipments of units, compare...
0.000	Lenovo IdeaPad Yoga 11: Hands-on with the bendy Windows RT tablet -
-0.105	John Cusack says Lenovo overtakes HP
-0.507	Lenovo Yoga Transforming Laptop Arrives, With Friends
-1.894	Lady's Black Laptop Bag for 15.6 inch Lenovo G560-0679AKU Notebook + An Ekatomi Hook. | Laptop Cases 15.6
-3.263	Lenovo IBM 0764 Series Notebook / Laptop Battery 5000mAh (Replacement): Lenovo IBM 0764 Series Notebook / Laptop...

scores them appropriately.

Two non-relevant reviews of Lenovo laptops have been given scores of 0.0 and -0.507. By themselves, these tweets are unlikely to be newsworthy, and the scores could be considered appropriate. Our approach does give a slightly negative score to one relevant tweet ("John Cusack says Lenovo overtakes HP"), however the tweet is lacking information and without context it would be extremely difficult to infer the newsworthiness of the story from only the tweet text.

Note that the same text appears twice in Table 5.12: "HP, Lenovo battle for top spot in PC market - Computerworld". Although the text is identical, these are distinct tweets, posted by different users at different times, and given different scores (2.177 / 2.070). The difference in scores is due to the responsive nature of our newsworthiness scoring approach. As new tweets are processed, the models are updated in real-time, allowing for new information to be incorporated and used to score subsequent tweets, resulting in the score difference.

5.4.8 Term Models

Although unigram terms are used for all experiments, it is interesting to examine how the use of bigrams affect performance. Table 5.13 shows how unigrams and bigrams performed using our standard test thresholds of 4.0 and 0.25 for LQ and HQ models respectively. Bigrams consistently underperform compared to the unigram models across all quality threshold scores tested.

Table 5.13: Newsworthiness classifications for Event and Other tweets using Unigram and Bigram term models.

	Newsworthy		Noise	
Model	**Event**	**Other**	**Event**	**Other**
Unigram	76.504%	18.830%	14.567%	50.133%
Bigrams	74.775%	20.829%	12.715%	54.677%

Table 5.14 shows some of the most frequent unigrams and bigrams for both the HQ and LQ models, filtered to remove any terms with likelihood ratios of less than 2.0. Many of the high quality bigrams refer to specific incidents or events, such as 'peace prize' or the infamous "Binders full of women" phrase used by Mitt Romney during the second U.S. Presidential Debate of 2012. This suggests that perhaps the bigram model is over-fitting and, rather than learning a generalizable scoring model, it is learning event-specific newsworthy phrases. Unigrams, on the other hand, are considerably more general, and tend to be verbs associated with common newsworthy event types, such as 'arrested' or 'shot'. The low quality unigrams and bigrams are semantically more similar to each other, and are often associated with games, marketing and publishing.

Table 5.14: The most frequent unigrams and bigrams with likelihood ratios of 2.0 or greater for both the HQ and LQ models.

	Unigrams
High Quality	has, says, win, check, read, vote, set, takes, join, using, writing, breaking, wins, killed, shows, closed, leads, arrested, shot, update, expected, lead, create, add, hits
Low Quality	completed, earned, joined, laughed, received, signed, upgrade, collected, androidgames, led, harvesting, collect, harvested, discover, correct, submitted, increase, reviews, includes, published, mount, provides, provided, determine, marketing
	Bigrams
High Quality	tribute to, peace prize, top news, us your, to lead, obama campaign, binders full, for its, courtesy of, this years, post on, south africa, in india, on october, at 35, new single, talks about, says he, 1 day, the election, this weeks, heres a, new video, presidential debate, fair and
Low Quality	step towards, pm rageofbahamut, the tribez, a member, surveys cant, 2 months, i earned, quest in, in valor, just completed, and is, this made, doing surveys, is making, club my, far from, android androidgames, androidgames gameinsight, made me, the club, completed the, rageofbahamut rageofbahamut, more look, for more, so far

5.5 Newsworthiness as a Feature for Event Detection

One of the aims of this work is to determine if newsworthiness can be incorporated into an existing event detection approach and whether it can be used as an feature to improve effectiveness. In this section, we examine how newsworthiness can be combined with a simple entity-based approach and used to filter out noisy clusters or clusters with a low Newsworthiness Score, improving event detection effectiveness.

The entity-based event detection approach described in chapter 4 provides a reasonable base approach which can be built upon. Since the aim is to evaluate the effectiveness of our Newsworthiness Score as a feature to identify newsworthy content, we ignore all non-clustering features used by the entity-based approach, such as entropy and burst detection. Without burst detection there is no way to combine clusters into larger events, so we report each cluster as an individual event.

Cluster Newsworthiness

We define the Newsworthiness Score N of cluster c as the mean Newsworthiness Score of each tweet d in the cluster:

$$N_c = \frac{\sum N_d}{D}$$

where D is the total number of tweets in the cluster.

5.5.1 Evaluation

Cosine is used to measure the distance between tweets, and each tweet is clustered with its nearest neighbour within a distance of 0.5. Although stopword removal and stemming are not used to calculate Newsworthiness Scores, we do stem and remove stopwords for clustering.

Clustering is performed in a streaming manner, and individual tweet newsworthiness is calculated in real-time, however mean cluster newsworthiness is calculated post-hoc, so results presented here do not represent a true streaming situation. However, the aim is to demonstrate that newsworthiness can be used as an effective feature for event detection, rather than to develop a novel real-time event detection technique. In practice, calculating the mean Newsworthiness Score for a cluster in real-time is trivial, and any difference in performance is likely to be small, however we note the difference for accuracy.

As we evaluate individual clusters as separate events, precision values reported here are likely to be lower than if temporally and topically similar clusters were combined into larger events as in chapter 4. This is due to an increase in the overall number of clusters evaluated, which increases the likelihood that a newsworthy and event-relevant cluster will evaluated as non-relevant. This is an unfortunate side-effect of how relevance judgements for the Events 2012 corpus were created, and is difficult to overcome without the use of a crowdsourced or manual evaluation by the authors.

Results

Table 5.15 shows Precision and Recall values at for clusters of various minimum Newsworthiness Scores and sizes. The top row, marked Any, shows baseline figures without any restrictions on cluster Newsworthiness, while other rows show precision and recall values after removing any clusters with Newsworthiness Scores below the filter value.

Table 5.15: Newsworthiness scores using different term models for tweets known to be relevant to a newsworthy event against the average score for all other tweets in the collection.

	5 Tweets		30 Tweets		50 Tweets		100 Tweets	
Score	**P**	**R**	**P**	**R**	**P**	**R**	**P**	**R**
Any	0.010	0.755	0.053	0.587	0.075	0.508	0.121	0.399
≥ 0	0.016	0.715	0.093	0.540	0.134	0.464	0.220	0.354
≥ 1	0.019	0.690	0.108	0.518	0.156	0.439	0.254	0.326
≥ 2	0.023	0.607	0.123	0.439	0.175	0.356	0.290	0.243
≥ 3	0.033	0.466	0.154	0.298	0.220	0.227	0.366	0.136
≥ 4	0.040	0.132	0.248	0.055	0.374	0.030	0.585	0.020

A clear trend can be see as both the minimum cluster size and minimum cluster Newsworthiness Score are increased; recall drops, but precision increases. With a minimum cluster size of 5, a maximum recall of 0.755 is achieved, however measured precision is very low at only 0.010. As we increase the minimum cluster Newsworthiness Score, recall drops off, however precision increases significantly. Although F1 scores are not shown, increasing the minimum Newsworthiness score towards 3 increases the F1 measure for all cluster sizes, however after 3, overall F1 score begins to decrease.

Although a fully crowdsourced evaluation would provide a detailed picture of how our Newsworthiness Scoring approach performs when used as a feature for event detection, it goes beyond the scope of this chapter. Instead, we perform a small manual evaluation focused on determining the precision of our Newsworthiness Scoring approach. We chose to evaluate clusters using a similar methodology to that used to create the Events 2012 corpus, however used only a single expert rather than 5 crowdsourced workers. We selected 2 sets of clusters to evaluate:

- 100 randomly selected clusters (of 4,270 total) with 5 or more tweets and a cluster Newsworthiness Score ≥ 4.
- All 115 clusters with 50 or more tweets and a cluster Newsworthiness score ≥ 4.

Of the 100 randomly selected clusters from 4,270 with 5 or more tweets, 95/100 clusters were marked as a significantly real-world event, giving a precision of 0.950. Of the 115 clusters with 50 or more tweets, all 115 were marked as significant real-world events, giving a precision of 1.0. Both of these results far exceed the automatically calculated precision values of 0.040 and 0.347 respectively. The reasons for this are similar to those described in chapter 4; although the collection has relevance judgements for a relatively large number of events, it does not cover all events during the month long

period the collection covers, or even all tweets for the 506 events it has judgements for.

These manually obtained results vastly outperform even the crowdsourced results presented in chapter 4 and suggest that Newsworthiness could be used as an extremely effective noise filter, requiring only 5 tweets for extremely high precision, and obtaining perfect precision at only 30 tweets. The automatically measured performance values are comparable to that of our entity-based event detection approach, despite being considerably simpler and being disadvantaged due to our use of clusters for evaluation rather than full events, as noted in section 5.5.1.

5.6 Conclusion

In this chapter, we proposed a novel newsworthiness scoring approach that uses a set of heuristics to automatically label content and learn term likelihood ratios to produce Newsworthiness Scores for tweets in real-time. We evaluated its classification and scoring effectiveness on the Events 2012 corpus and found that it was able to distinguish between content related to real-world events and noise. Automatic evaluation on the Events 2012 corpus showed that by using Newsworthiness as a feature to filter out noisy clusters, we could significantly improve the effectiveness of a simple cluster-based event detection approach, achieving significant improvements to precision with only slight decreases to recall. Finally, a manual evaluation suggests that high newsworthiness scores can be extremely effective at filtering out noise, and we achieved precision values of 0.950 on clusters as small as 5 tweets, and perfect precision with as few as 50.

CHAPTER 6

Conclusion

We conclude with a summary of each chapter and detail their main contributions. We then examine our findings in relation to the research questions set out in chapter 1. Finally, we describe future areas and directions of research.

In chapter 2 we described the background information necessary to understand this thesis. We gave an overview of Information Retrieval, described the Topic Detection and Tracking project, and summarised the most relevant literature.

In chapter 3 we described the creation of the first large-scale corpus for the evaluation of event detection approaches on Twitter. We examined existing definitions of 'event' and proposed a new definition refined for event detection on Twitter. We detailed the approaches used to generate candidate events, and the crowdsourced methodology used to gather annotations and relevance judgements.

In chapter 4 we proposed a novel entity-based event detection approach for Twitter, that uses named entities (people, places, organizations, etc.) to partition and efficiently cluster tweets. We performed an in-depth evaluation of the detection approach using the Events 2012 corpus, which we believe was the first of its kind, and compared automated evaluation approaches with a crowdsourced evaluation. We described some of the issues that remain to be solved before automated evaluation can full replace crowdsourced evaluations of event detection approaches.

Finally, in chapter 5, we proposed a method of scoring tweets in real-time based on their Newsworthiness. We used heuristics to assign quality labels to tweets and learn term likelihood ratios, and calculate Newsworthiness scores. We evaluated the classification and scoring accuracy using the Events 2012 corpus, and found it to be effective at classifying documents as Newsworthy or Noise. We proposed a cluster Newsworthiness score that can be used as a feature for event detection, and evaluated it by filtering clusters produced using the entity-based clustering approach proposed in chapter 4. We found that cluster Newsworthiness can be used to signification increase precision even at small cluster sizes.

6.1 Research Questions

RQ1: Can we develop a methodology that allows us to build a test collection for the evaluation of event detection approaches on Twitter?

In chapter 3, we answered this research question by creating a large-scale corpus with relevance judgements for the evaluation of event detection on Twitter. Since the publication of McMinn et al. [2013] describing the corpus, more than 240 researchers and groups have registered to download the Events 2012 corpus, and it has been cited by more than 90 publications, and used in the development and evaluation of several event detection approaches for Twitter (including several PhD and Masters theses). We used the collection we developed to evaluate our entity-based event detection approach, and our newsworthiness scoring technique, demonstrating that the collection is suitable for evaluating event detection approaches on Twitter.

RQ2: Can entities (people, places, organizations) be used to detect real-world events in a streaming setting on Twitter?

Chapter 4 describes our proposed entity-based, real-time event detection approach for Twitter. Our entity-based approach partitions tweets based on the entities they contain to perform real-time clustering in an efficient manner, and uses a light-weight burst detection approach to identify unusual volumes of discussion around entities. We found that it is possible to use entities to detect real-world event in a streaming setting on Twitter, and by evaluating this approach using the Events 2012 corpus, we found that it out-performed two state-of-the-art baselines.

RQ3: How fairly and accurately can automatic evaluation approaches evaluate event detection approaches for Twitter?

In chapter 4, we used an automated evaluation methodology to evaluate our proposed event detection approach, and examined how these results compare to a crowdsourced evaluation. We determined that although it is possible to automatically evaluate event detection approaches for Twitter, there remain a number of key challenges and issues that need to be addressed before automated evaluation can fully replace manual evaluation of event detection approaches. Hasan et al. [2017] surveyed real-time event detection techniques for Twitter in early 2017, and noted that the Events 2012 corpus remained the only corpus for the evaluation of event detection approaches on Twitter, suggesting that its continued use could help conduct fair performance comparisons between different event detection approaches [Hasan et al. 2017].

RQ4: Can we determine the newsworthiness of an individual tweet from content alone?

The Newsworthiness Scoring approach we developed in chapter 5 uses a set of heuristics to assign quality labels to tweets and learn term likelihood ratios to produce Newsworthiness Score for tweets in real-time. We evaluated the scores as a classification and scoring task, and found that the approach is able to label Newsworthy and Noise tweets with a high degree of accuracy. We then used the Newsworthiness Score to estimate cluster Newsworthiness as a feature for event detection. We used the entity-based clustering approach proposed in chapter 4, but filtered out clusters with low newsworthiness scores, resulting in extremely high precision with very few tweets.

6.2 Future Work

There are a number of ways that the work in this thesis could be extended, the most obvious and pressing of which is how to improve the automatic evaluation of event detection approaches, and to determine how measures can be improved to better evaluate the precision of detection approaches. The Newsworthiness Score presented in chapter 5 shows much promise, and could easy be integrated into more detection approaches, or perhaps used as the basis for a new event detection approach.

Bibliography

Younos Aboulnaga, Charles L. A. Clarke, and David R. Cheriton. Frequent itemset mining for query expansion in microblog ad-hoc search.

Charu Aggarwal and Karthik Subbian. Event detection in social streams. In *SDM*, pages 624–635. SIAM / Omnipress, 2012. ISBN 978-1-61197-232-0.

James Allan. Topic detection and tracking. chapter Introduction to topic detection and tracking, pages 1–16. Kluwer Academic Publishers, Norwell, MA, USA, 2002a. ISBN 0-7923-7664-1. URL http://dl.acm.org/citation.cfm?id=772260.772262.

James Allan. *Topic detection and tracking: event-based information organization*, volume 12. Springer, 2002b.

James Allan, Victor Lavrenko, and Hubert Jin. First story detection in TDT is hard. CIKM '00, pages 374–381, New York, NY, USA, 2000a. ACM. ISBN 1-58113-320-0. doi: 10.1145/354756.354843. URL http://doi.acm.org/10.1145/354756.354843.

James Allan, Victor Lavrenko, Daniella Malin, and Russell Swan. Detections, bounds, and timelines: UMass and TDT-3. In *TDT-3 Workshop*, 2000b.

James Allan, Stephen Harding, David Fisher, Alvaro Bolivar, Sergio Guzman-Lara, and Peter Amstutz. Taking topic detection from evaluation to practice. In *HICSS'05*, HICSS '05, Washington, USA, 2005. IEEECS. ISBN 0-7695-2268-8-4. doi: 10.1109/HICSS.2005.576. URL http://dx.doi.org/10.1109/HICSS.2005.576.

Omar Alonso, Daniel E. Rose, and Benjamin Stewart. Crowdsourcing for relevance evaluation. *SIGIR Forum*, 42(2):9–15, November 2008. ISSN 0163-5840. doi: 10.1145/1480506.1480508. URL http://doi.acm.org/10.1145/1480506.1480508.

G. Amati, G. Amodeo, M. Bianchi, A. Celi, C.D. Nicola, M. Flammini, C. Gaibisso, G. Gambosi, and G. Marcone. Fub, iasi-cnr, univaq at trec 2011. In *Text REtrieval Conference (TREC 2011)*. US, 2011.

Gianni Amati and Cornelis Joost Van Rijsbergen. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Trans. Inf. Syst.*, 20(4):357–389, October 2002. ISSN 1046-8188. doi: 10.1145/582415.582416. URL http://doi.acm.org/10.1145/582415.582416.

Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Inf. Retr.*, 12 (4):461–486, August 2009. ISSN 1386-4564. doi: 10.1007/s10791-008-9066-8. URL http://dx.doi.org/10.1007/s10791-008-9066-8.

Farzindar Atefeh and Wael Khreich. A survey of techniques for event detection in twitter. *Comput. Intell.*, 31(1):132–164, February 2015. ISSN 0824-7935. doi: 10.1111/coin.12017. URL http://dx.doi.org/10.1111/coin.12017.

Amit Bagga and Breck Baldwin. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98, pages 79–85, Stroudsburg, PA, USA, 1998. Association for Computational Linguistics. doi: 10.3115/980845.980859. URL http://dx.doi.org/10.3115/980845.980859.

Hila Becker, Mor Naaman, and Luis Gravano. Learning similarity metrics for event identification in social media. WSDM '10, pages 291–300, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-889-6. doi: 10.1145/1718487.1718524. URL http://doi.acm.org/10.1145/1718487.1718524.

Hila Becker, Mor Naaman, and Luis Gravano. Beyond trending topics: Real-world event identification on twitter. WISDM'11, 2011a.

Hila Becker, Mor Naaman, and Luis Gravano. Beyond trending topics: Real-world event identification on twitter. In *ICWSM'11*, ICWSM'11, 2011b.

Hila Becker, Dan Iter, Mor Naaman, and Luis Gravano. Identifying content for planned events across social media sites. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, pages 533–542, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-0747-5. doi: 10.1145/2124295.2124360. URL http://doi.acm.org/10.1145/2124295.2124360.

Fabrıcio Benevenuto, Gabriel Magno, Tiago Rodrigues, and Virgılio Almeida. Detecting spammers on twitter. In *Collaboration, electronic messaging, anti-abuse and spam conference (CEAS)*, volume 6, 2010.

D. Boyd, S. Golder, and G. Lotan. Tweet, tweet, retweet: Conversational aspects of retweeting on twitter. In *System Sciences (HICSS), 2010 43rd Hawaii International Conference on*, pages 1–10, 2010. doi: 10.1109/HICSS.2010.412.

Chris Buckley, Darrin Dimmick, Ian Soboroff, and Ellen Voorhees. Bias and the limits of pooling for large collections. *Inf. Retr.*, 10(6):491–508, December 2007. ISSN 1386-4564. doi: 10.1007/s10791-007-9032-x. URL http://dx.doi.org/10.1007/s10791-007-9032-x.

Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. Information credibility on twitter. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 675–684, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0632-4. doi: 10.1145/1963405.1963500. URL http://doi.acm.org/10.1145/1963405.1963500.

Moses S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the Thiry-fourth Annual ACM Symposium on Theory of Computing*, STOC '02, pages 380–388, New York, NY, USA, 2002. ACM. ISBN 1-58113-495-9. doi: 10.1145/509907.509965. URL http://doi.acm.org/10.1145/509907.509965.

Smitashree Choudhury and John G. Breslin. Extracting semantic entities and events from sports tweets. In *In Proceedings of the 1st Workshop on Making Sense of Microposts (#MSM2011) at ESWC*, pages 22–32, 2011.

Leon Derczynski, Alan Ritter, Sam Clark, and Kalina Bontcheva. Twitter part-of-speech tagging for all: Overcoming sparse and noisy data. In *ICRA-NLP '13'*. ACL, 2013.

Paul Ferguson, Neil OHare, James Lanagan, Alan F Smeaton, Owen Phelan, Kevin McCarthy, and Barry Smyth. Clarity at the trec 2011 microblog track. 2011.

M. Forelle, P. Howard, A. Monroy-Hernández, and S. Savage. Political Bots and the Manipulation of Public Opinion in Venezuela. *ArXiv e-prints*, July 2015.

Anuradha Goswami and Ajey Kumar. A survey of event detection techniques in online social networks. *Social Network Analysis and Mining*, 6(1): 107, Nov 2016. ISSN 1869-5469. doi: 10.1007/s13278-016-0414-1. URL https://doi.org/10.1007/s13278-016-0414-1.

Chris Grier, Kurt Thomas, Vern Paxson, and Michael Zhang. @spam: The underground on 140 characters or less. In *ACM CCS '10'*, CCS '10, pages 27–37, New York,

NY, USA, 2010. ACM. ISBN 978-1-4503-0245-6. doi: 10.1145/1866307.1866311. URL http://doi.acm.org/10.1145/1866307.1866311.

Alper Gün and Pınar Karagöz. A hybrid approach for credibility detection in twitter. In Marios Polycarpou, André C. P. L. F. de Carvalho, Jeng-Shyang Pan, Michał Woźniak, Héctor Quintian, and Emilio Corchado, editors, *Hybrid Artificial Intelligence Systems*, pages 515–526, Cham, 2014. Springer International Publishing. ISBN 978-3-319-07617-1.

Zhongyuan Han, Xuwei Li, Muyun Yang, Haoliang Qi, Sheng Li, and Tiejun Zhao. Hit at trec 2012 microblog track.

Mahmud Hasan, Mehmet A Orgun, and Rolf Schwitter. A survey on real-time event detection from the twitter data stream. *Journal of Information Science*, 2017. doi: 10.1177/0165551517698564. URL https://doi.org/10.1177/0165551517698564.

P. N. Howard and B. Kollanyi. Bots, #StrongerIn, and #Brexit: Computational Propaganda during the UK-EU Referendum. *ArXiv e-prints*, June 2016.

Mengdie Hu, Shixia Liu, Furu Wei, Yingcai Wu, John Stasko, and Kwan-Liu Ma. Breaking news on twitter. CHI '12, pages 2751–2754, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1015-4. doi: 10.1145/2208636.2208672. URL http://doi.acm.org/10.1145/2208636.2208672.

David Jurgens and Keith Stevens. Event detection in blogs using temporal random indexing. eETTs '09, pages 9–16, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-954-452-011-3. URL http://dl.acm.org/citation.cfm?id=1859650.1859652.

Byungkyu Kang, John O'Donovan, and Tobias Höllerer. Modeling topic specific credibility on twitter. In *Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces*, IUI '12, pages 179–188, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1048-2. doi: 10.1145/2166966.2166998. URL http://doi.acm.org/10.1145/2166966.2166998.

Yubin Kim, Reyyan Yeniterzi, and Jamie Callan. Overcoming vocabulary limitations in twitter microblogs.

B. Klimt and Y. Yang. Introducing the Enron corpus. In *First Conference on Email and Anti-Spam (CEAS)*, 2004.

Giridhar Kumaran and James Allan. Text classification and named entities for new event detection. SIGIR '04, pages 297–304, New York, NY, USA, 2004. ACM. ISBN 1-58113-881-4. doi: 10.1145/1008992.1009044. URL http://doi.acm.org/10.1145/1008992.1009044.

Giridhar Kumaran and James Allan. Using names and topics for new event detection. In *HLT '05'*, HLT '05, pages 121–128, Stroudsburg, PA, USA, 2005. ACL. doi: 10.3115/1220575.1220591. URL http://dx.doi.org/10.3115/1220575.1220591.

Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? WWW '10, pages 591–600, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-799-8. doi: 10.1145/1772690.1772751. URL http://doi.acm.org/10.1145/1772690.1772751.

Chenliang Li, Jianshu Weng, Qi He, Yuxia Yao, Anwitaman Datta, Aixin Sun, and Bu-Sung Lee. Twiner: named entity recognition in targeted twitter stream. In *SIGIR*, pages 721–730, 2012a.

Q. Li, A. Nourbakhsh, S. Shah, and X. Liu. Real-time novel event detection from social media. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, pages 1129–1139, April 2017. doi: 10.1109/ICDE.2017.157.

Rui Li, Kin Hou Lei, Ravi Khadiwala, and Kevin Chen-Chuan Chang. Tedas: A twitter-based event detection and analysis system. *Data Engineering, International Conference on*, 0:1273–1276, 2012b. ISSN 1084-4627. doi: http://doi.ieeecomputersociety.org/10.1109/ICDE.2012.125.

Yan Li, Zhenhua Zhang, Wenlong Lv, Qianlong Xie, Yuhang Lin, Rao Xu, Weiran Xu, Guang Chen, and Jun Guo. Pris at trec2011 micro-blog track. In *Text REtrieval Conference Proceedings. NIST*, 2011.

P. K. K. Madhawa and A. S. Atukorale. A robust algorithm for determining the newsworthiness of microblogs. In *2015 Fifteenth International Conference on Advances in ICT for Emerging Regions (ICTer)*, pages 135–139, Aug 2015. doi: 10.1109/ICTER.2015.7377679.

Richard McCreadie, Ian Soboroff, Jimmy Lin, Craig Macdonald, Iadh Ounis, and Dean McCullough. On building a reusable twitter corpus. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '12, pages 1113–1114, New York, NY, USA,

2012. ACM. ISBN 978-1-4503-1472-5. doi: 10.1145/2348283.2348495. URL http://doi.acm.org/10.1145/2348283.2348495.

Andrew J. McMinn and Joemon M. Jose. Real-time entity-based event detection for twitter. In Josanne Mothe, Jacques Savoy, Jaap Kamps, Karen Pinel-Sauvagnat, Gareth Jones, Eric San Juan, Linda Capellato, and Nicola Ferro, editors, *Experimental IR Meets Multilinguality, Multimodality, and Interaction*, pages 65–77, Cham, 2015. Springer International Publishing. ISBN 978-3-319-24027-5.

Andrew J. McMinn, Yashar Moshfeghi, and Joemon M. Jose. Building a large-scale corpus for evaluating event detection on twitter. In *Proceedings of the 22Nd ACM International Conference on Conference on Information & Knowledge Management*, CIKM '13, pages 409–418, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2263-8. doi: 10.1145/2505515.2505695. URL http://doi.acm.org/10.1145/2505515.2505695.

Andrew J. McMinn, Daniel Tsvetkov, Tsvetan Yordanov, Andrew Patterson, Rrobi Szk, Jesus A. Rodriguez Perez, and Joemon M. Jose. An interactive interface for visualizing events on twitter. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '14, pages 1271–1272, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2257-7. doi: 10.1145/2600428.2611189. URL http://doi.acm.org/10.1145/2600428.2611189.

Thin Nguyen, Dinh Phung, Brett Adams, and Svetha Venkatesh. Emotional reactions to real-world events in social networks. PAKDD'11, pages 53–64, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 978-3-642-28319-2.

Chaluemwut Noyunsan, Tatpong Katanyukul, Yuqing Wu, and Kanda Runapongsa Saikaew. A social network newsworthiness filter based on topic analysis. *International Journal of Technology*, 7(7), 2016. ISSN 2087-2100.

Chaluemwut Noyunsan, Tatpong Katanyukul, Carson K. Leung, and Kanda Runapongsa Saikaew. Effects of the inclusion of non-newsworthy messages in credibility assessment. In Grigori Sidorov and Oscar Herrera-Alcántara, editors, *Advances in Computational Intelligence*, pages 185–195, Cham, 2017. Springer International Publishing. ISBN 978-3-319-62434-1.

M. Osborne, S. Petrovic, R. McCreadie, C. Macdonald, and I. Ounis. Bieber no more: First Story Detection using Twit-

ter and Wikipedia. *SIGIR 2012 Workshop TAIA*, 2012. URL http://www.dcs.gla.ac.uk/ craigm/publications/osborneTAIA2012.pdf.

Ozer Ozdikis, Pinar Senkul, and Halit Oguztüzün. Semantic expansion of tweet contents for enhanced event detection in twitter. In *ASONAM*, pages 20–24. IEEE Computer Society, 2012. ISBN 978-0-7695-4799-2.

Saša Petrović, Miles Osborne, and Victor Lavrenko. Streaming first story detection with application to twitter. In *HLT '10'*, HLT '10, pages 181–189, Stroudsburg, PA, USA, 2010a. Association for Computational Linguistics. ISBN 1-932432-65-5. URL http://dl.acm.org/citation.cfm?id=1857999.1858020.

Saša Petrović, Miles Osborne, and Victor Lavrenko. Streaming first story detection with application to twitter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 181–189, Stroudsburg, PA, USA, 2010b. Association for Computational Linguistics. ISBN 1-932432-65-5. URL http://dl.acm.org/citation.cfm?id=1857999.1858020.

Saša Petrović, Miles Osborne, and Victor Lavrenko. Using paraphrases for improving first story detection in news and twitter. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '12, pages 338–346, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. ISBN 978-1-937284-20-6. URL http://dl.acm.org/citation.cfm?id=2382029.2382072.

Friedrich Pukelsheim. The three sigma rule. *The American Statistician*, 48(2):pp. 88–91, 1994. ISSN 00031305. URL http://www.jstor.org/stable/2684253.

Justus J Randolph, A Thanks, R Bednarik, and N Myller. Free-marginal multirater kappa (multirater κfree): an alternative to fleiss' fixed-marginal multirater kappa. In *Joensuu learning and instruction symposium*. Citeseer, 2005.

Alan Ritter, Mausam, Oren Etzioni, and Sam Clark. Open domain event extraction from twitter. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 1104–1112, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1462-6. doi: 10.1145/2339530.2339704. URL http://doi.acm.org/10.1145/2339530.2339704.

Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. WWW '10, NY, USA,

2010. ACM. ISBN 978-1-60558-799-8. doi: 10.1145/1772690.1772777. URL http://doi.acm.org/10.1145/1772690.1772777.

Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, 24(5):513–523, August 1988. ISSN 0306-4573. doi: 10.1016/0306-4573(88)90021-0. URL http://dx.doi.org/10.1016/0306-4573(88)90021-0.

Gerard Salton, Anita Wong, and Chung-Shu Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

Jagan Sankaranarayanan, Hanan Samet, Benjamin E Teitler, Michael D Lieberman, and Jon Sperling. Twitterstand: news in tweets. In *ACM SIGSPATIAL '09'*. ACM, 2009.

C. E. Shannon. A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5(1):3–55, January 2001. ISSN 1559-1662. doi: 10.1145/584091.584093. URL http://doi.acm.org/10.1145/584091.584093.

Ka Cheung Sia, Junghoo Cho, Yun Chi, and Belle L. Tseng. Efficient computation of personal aggregate queries on blogs. KDD '08, pages 632–640, NY, USA, 2008. ACM. ISBN 978-1-60558-193-4. doi: 10.1145/1401890.1401967. URL http://doi.acm.org/10.1145/1401890.1401967.

S. Sikdar, B. Kang, J. ODonovan, T. Höllerer, and S. Adah. Understanding information credibility on twitter. In *2013 International Conference on Social Computing*, pages 19–24, Sept 2013a. doi: 10.1109/SocialCom.2013.9.

Sujoy Kumar Sikdar, Byungkyu Kang, John O'Donovan, Tobias Hollerer, and Sibel Adal. Cutting through the noise: Defining ground truth in information credibility on twitter. *Human*, 2(3):151–167, 2013b.

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

Yiming Yang, Tom Pierce, and Jaime Carbonell. A study of retrospective and on-line event detection. SIGIR '98, pages 28–36, New York, NY, USA, 1998. ACM. ISBN 1-58113-015-5. doi: 10.1145/290941.290953. URL http://doi.acm.org/10.1145/290941.290953.

Qiankun Zhao, Prasenjit Mitra, and Bi Chen. Temporal and information flow based event detection from social text streams. AAAI'07,

pages 1501–1506. AAAI Press, 2007. ISBN 978-1-57735-323-2. URL http://dl.acm.org/citation.cfm?id=1619797.1619886.

Xin Zhao and Jing Jiang. An empirical comparison of topics in twitter and traditional media. *Singapore Management University School of Information Systems Technical paper series.*, 10:2011, 2011.

Justin Zobel. How reliable are the results of large-scale information retrieval experiments? In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98, pages 307–314, New York, NY, USA, 1998. ACM. ISBN 1-58113-015-5. doi: 10.1145/290941.291014. URL http://doi.acm.org/10.1145/290941.291014.