SocialStories: Segmenting Stories within Trending Twitter Topics

Kokil Jaidka
Big Data Experience Lab
Adobe Systems Pvt Ltd
Bengaluru, India
jaidka@adobe.com

Kaushik Ramachandran
Adobe Systems Pvt Ltd
Bengaluru India
rkaushi@adobe.com

Prakhar Gupta
Big Data Experience Lab
Adobe Systems Pvt Ltd
Bengaluru, India
prakhgup@adobe.com

Sajal Rustagi
Indian Institute of Technology
Roorkee, India
sajalrustagi1993@gmail.com

ABSTRACT

This study present SocialStories - a system based on incremental clustering for streaming tweets, for identifying fine-grained stories within a broader trending topic on Twitter. The contributions include a novel tf-metric, called the inverse cluster frequency, and a decay weighting for entities. We present our experiments on 0.19 million tweets posted in June 2014, revolving around the mentions of a software brand before, during and after a marketing conference and a software release. The novelty of our work is the text-based similarity calculation metrics, including a new similarity metric, called the inverse cluster frequency, and time-specific metrics that allow for the decay of old entities with the passage of time and preserve the homogeneity and the freshness of themes. We report improved performance and higher recall of 80%, against the gold standard (posthoc journalistic reports), as compared to LDA-, and Wavelet-based systems. Our algorithm is able to cluster 80% of all tweets into story-based clusters, which are 86% pure. It also enables earlier detection of trending stories than manual reports, and is far more accurate in identifying fine-grained stories within sub-topics as compared to baseline systems.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering, Selection process, Clustering*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*text analysis*

General Terms

Algorithms, Experimentation

Keywords

Topic Detection, Text Classification, Information Retrieval, Clustering, NLP, text analysis, document categorization, topic labelling, social media, Twitter, trending topics, tweet themes

1. INTRODUCTION

This work addresses the twin problems of detecting new stories and identifying story boundaries in social media discussions. Micro-blogging platforms like Twitter act as a platform for users worldwide to create content and consume information and share messages with each other or with the larger community. [1] report that there are 500 million active users on Twitter at any given minute, and millions of tweets are posted every second. This makes Twitter a powerful and timely source of information about breaking news, international events and the general topics in popular discourse.

Twitter's stream of tweets comprises unmoderated and unverified 'tweets' posted by millions of users, which discuss several topics, each of which comprises a set of unique 'stories' [2]. Topics are built upon a triggering event, such as a product release or a country's election. They are delimited in scope - that is, they begin at a set time and will probably no longer be discussed after some time. Stories are facets to a broader topic, and are identified using the content and timeline features of posts. For example, in the case of a product release, there would be separate stories to discuss the product, related products and the parent brand and the speakers in the launch event. In the case of a conference, stories could discuss the venue, sponsors, sessions, the keynote speech, speakers, even the weather, the general arrangements or the milieu. However, the volume of information make it overwhelming for a user looking for specific information or interested in tracking one particular topic over time, and creates a filter failure problem [15]. Further, in browsing from post to post, it is not possible for users to quickly drill in to aggregate information, into posts about certain themes, and then drill back out to the aggregate information to understand the broader topic more vividly.

Solving the story segmentation and first story detection problem for social media is challenging for several reasons. Every tweet can be seen as a list of words that can link it to the story it discusses. However, tweets are delimited by length - they can be a maximum of 140 characters in length, which makes it difficult to assign them into appropriately distinct categories. Furthermore, tweets are not simply sentences, but contain other components such as mentions, hyperlinks and hashtags. In particular, we are interested in the 'entities' which are explicitly mentioned, or implicitly referenced, in tweets - these may be noun phrases representing the names of people, places, organizations and events, or they may be hashtags, which are keywords prefixed with the '#' character. Another challenge with identifying stories on social media is that the tremendous volume of tweets per second makes it a difficult problem to solve in real-time, and the timely nature of social media means that posts have a short lifespan in which they may be current or relevant to a story, after which the posts may decay, and the conversation may digress in a different direction. Finally, the variety and noise in the posts make it challenging to solve topic tracking problems in real-time.

In our approach, we have identified and extracted new text- and time-based features in an incremental clustering method, which compares incoming posts against existing stories, and detects new stories as they emerge. The novelty of our work is the text-based similarity calculation metrics, including a new similarity metric, called the inverse cluster frequency, and time-specific metrics that allow for the decay of old entities with the passage of time and preserve the homogeneity and the freshness of themes.

2. RELATED WORK

This study is conducted in the context of the Topic Detection and Tracking problem for Twitter, which considers a constantly arriving stream of text. Topic detection and tracking is a five-stage problem, comprising first story detection, story segmentation, cluster detection, story tracking and story link detection [2]. In this work, we address story segmentation in Twitter, which refers to the problem of dividing the incoming streaming Twitter posts into individual stories; as a part of this, we also propose a method for first story detection, which refers to the problem of recognizing the onset of a new story in streaming content.

Work in story segmentation in social media mainly focuses on identifying "bursty" themes - however, these approaches suffer from a high duplicate-event rate, which means that the same theme is detected more than once. [12] presented hashtag based schemes to improve topic modeling in microblogs, using an implementation of LDA, while [8] used dictionary learning in their clustering technique to cluster tweets. [14] experimented on clustering tweets based on their cosine similarity with keywords to group tweets using a supervised k-nearest neighbor approach. These works do not leverage the temporality of posts, which are an important and characteristic property of conversations in a social community. In our work, we develop our own clustering technique for segmenting themes based on how closely their content resembles, and is contained in, existing themes, while incorporating natural decay of old themes over time.

With regards to the story detection problem, Enblogue by [3] detects new emerging stories represented by tagged content, by calculating correlations among tags, to find new emerging tags and trends in Twitter. Other systems, such as Twitter Monitor by [11], do a frequency analysis of co-occurring words. However, this would mean that several stories occurring at the same time would be identified as a single story. [5] used content aging theory and formalized life cycle of keywords to determine trending and emerging stories based on the energy, nutrition value as well as importance of content based on social relationships in user network. However, these methods do not consider the cases when old stories may be identified as new trends if they show a spike in a short period of time, thus losing out on relevant content posted a short time back.

These works convey the idea that the two problems of identifying new themes, and segmenting existing themes, are interesting and important to solve however, existing approaches have not attempted to solve both problems together. Thus motivated, we offer an unsupervised incremental clustering approach which emulates the real-time nature of user generated posts in a social media discussion. The data considered for evaluation of cluster specific features at a particular time will consist of all documents from the stream with timestamps less than and equal to the last time windows end time. The method is based on how closely the incoming content resembles existing stories and incorporates natural decay of old stories over time.

3. THE SOCIALSTORIES FRAMEWORK

The SocialStories framework (Figure 1) is an unsupervised single pass incremental online clustering algorithm to cluster a stream of incoming tweets in real time. Given a list of tweets t1, t2, t3 ...tn, the algorithm takes the tweets one by one and computes their similarity with every existing cluster C_i. Similarity is a measure of the *containment* or importance of an entity to a story, *resemblance* or similarity between entities, and a novel tf-metric which we call the *inverse cluster frequency*, a measure of the uniqueness of an entity with respect to its presence in different tweet clusters. Every cluster represents a story, and new clusters are created when the maximum similarity falls below a parameter, which is set empirically in the training phase. The framework emulates the real-time nature of social media data; the data considered for calculating and updating at a particular time will consist of all documents from the stream with timestamps less than and equal to the last time window's end time. In the rest of this paper, "cluster" and "story" are used interchangeably to reference a group of tweets which discuss the same, fine-grained story within an overall, broad topic.

Step 1: Model posts as feature vectors - The framework

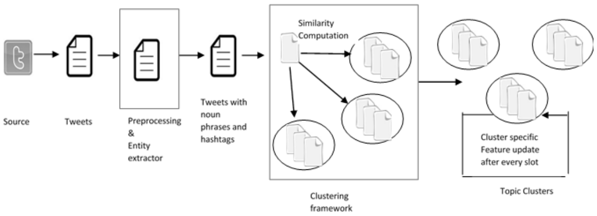
Source
Tweets
Preprocessing & Entity extractor
Tweets with noun phrases and hashtags
Similarity Computation
Clustering framework
Cluster specific Feature update after every slot
Topic Clusters

Figure 1: The SocialStories framework

is an unsupervised single pass incremental online clustering algorithm to cluster a stream of incoming tweets in real time. Incoming tweets are represented as feature vectors and compared against existing stories, where every story is a 'cluster', and is represented by the feature vector of its centroid. New themes are created when the maximum similarity falls below a parameter. The following procedure is used to obtain feature vectors:

1. Syntactic and semantic parsing to obtain text snippets, which will be nouns, noun phrases, and words of known semantic category such as persons names, organizations names, names of software products and so on. Certain frequent words (stop-words) are removed.

2. Initial weights are assigned to text snippets using the normalized term frequency of the post. Normalized term frequency is calculated as a proportion of the term (one particular text snippet) to total number of text snippets found in the post. This is done as follows:

   $$w(entity, tweet) = (1+log_2 tf_{(entity,tweet)}) \times icf_{entity}$$

   Here, $w(ent, tweet)$ is the weight of the entity within the tweet, $tf_(term, tweet)$ is the count of the entity within the tweet, and icf is the inverse cluster frequency after normalization.

3. Known keywords, entered by the user or preset by the system, which exist in the post or are otherwise tagged in association with the post, are identified and Boosted. This is because they can act as good indicators of the posts theme. The value of the boost weight can be set empirically or based on iterative updates. For a cluster centroid $C_(i)$://

   $$boostedtf_{entity} = boost \times normalizedtf_{entity}$$

   $$normalizedtf(entity, C_i) = 0.5 + (\frac{0.5 \times freq(entity, C_i)}{max\{freq(entity, C_i)\epsilon C_i\}})$$

   Here, $normalizedtf(entity, C_i)$ is the normalized term frequency for an entity, $boostedtf_{entity}$ is the boosted tf value of hashtags, and $freq(entity, C_i)$ is the frequency of an entity (including hashtags) in a cluster.

Step 2: Calculate Feature Weights -The following paragraphs discuss the similarity metrics used to assign a post to a theme.

Containment- The containment c(A;B) of A in B is a number between 0 and 1 which reflects how much of A is roughly contained within B. Containment accounts for the importance of a feature for the post. For textual features, as the number of text snippets in a post increases, the relevance of one entity for the post decreases. For example, taking the case of tweets mentioning computer software - if a tweet contains more than one entity like Adobe, Photoshop, Illustrator and Reader, the probability that tweet belongs to a cluster about Adobe Photoshop decreases. This is also one way in which our algorithm fights spam tweets, which may mention a lot of popular keywords without providing any real content.

$$Containment = c(entity, C_i) = \frac{|S(t) \cap S(C_i)|}{|S(t)|}$$

Where c(entity,C_i) denotes containment, S(t) denotes entities in a tweet and the numerator denotes set of entities in the tweet which were present in the cluster. The denominator is the union of all entities present in either the tweet or the cluster.

Resemblance - Resemblance assesses similarity between the post and existing themes. In our approach, we have calculated resemblance using Jaccard coefficient, which compares the intersection of features between the post and the theme against the union of features.

$$Resemblance = r(t, C_i) = \frac{|S(t) \cap S(C_i)|}{|S(t) \cup S(C_i)|}$$

Where r(t,C_i) denotes resemblance, S(t) denotes entities in a tweet and the numerator denotes set of entities in the tweet which were present in the cluster. The denominator is the union of all entities present in either the tweet or the cluster.

Inverse Cluster Frequency - Inverse cluster frequency shows the uniqueness of a feature within a theme. For textual features, it represents the defining characteristics of a theme. For user features, it identifies the dominant participants in a theme. The rationale behind this feature is that, posts belonging to different stories may share some textual snippets; on the other hand, text which is present in more than one story should receive lower weights than those which are unique to a story, because they will be more indicative of the theme contents. It is depicted in the formula below:

$$icf_{entity} = \frac{1 + log_{10}(\frac{TotalNoofclusters}{Noofclustersitispresent})}{max\{icf(e), e\epsilon Cluster\}}$$

Step 3 - Incremental Clustering Algorithm After a post t is represented as a vector of weighted entities, its similarity against existing clusters is calculated, using any similarity or distance metric. In this case, we have used the cosine similarity formula

$$sim(t, C_i) = \frac{\sum_{i=1}^{N} w_{i,t} w_{i,C_i}}{\sqrt{\sum_{i=1}^{N} w_{i,t}^2} \sqrt{\sum_{j=1}^{N} w_{i,C_i}^2}}$$

where the similarity sim(t,C_i) between the tweet and cluster centroid is calculated considering as a sum of the weights of all entities, both in the tweet and the cluster centroid. If a post is identified as belonging to a theme, then its new keywords are incorporated into the cluster centroid representing the theme.

Step 4: Timely Update and Decay - After cluster assignment, time-based features are used to gradually incrementally update the weight of entities over time, as well as gradually decay old entities and posts, for a theme.

Incremental Update of Feature Weights - Besides temporal decay, the tf values of every feature in a post is separately updated with the addition of every post in the theme. For the first k posts of a theme, the inverse-theme-frequency values are updated after the arrival of every n number of posts, so that the weights assigned to the initial features have the most recent values. As new posts enter in a cluster, the tf of any post feature present in the themes feature vector is updated by the overall similarity calculated with the theme, and multiplied by the weight of the feature in the post, and then stored in its normalized form; similarly the icf is updated too. For user features, update would be based on the new users participating in the post or theme, or the same user participating multiple times.

$$updatedtf_{entity,C_i} = oldtf_{entity,C_i} + weight(entity, t) \times sim(t, C_i)$$

Here, $oldtf(entity, C_i)$ and $updatedtf(entity, C_i)$ are the original and the updated term frequency for an entity. $weight(entity, t)$ is the weight of an entity within a tweet, $sim(t, C_i)$ is a measure of similarity between the entity distribution of the tweet and the cluster C_i.

Decay - This feature captures the temporal shift of the weight of features and adjusts their weight. In a theme which has been active for some time, consider a feature that was used from the time the theme was formed, and its term frequency value increased accordingly. But even if a new feature, included in the last few time slots, its term frequency value will be lower than the former. To keep track of the latest vocabulary, the algorithm will update values in a single timeslot, and the values gained by a keyword over a time slot is carried over to other time slots, but with a dampening factor.

$$decayedtf_{entity,C_i} = \sum_{t=t_{now}}^{t_{now-k}} e^{-t/T} \times tf_t$$

Here, $decayedtf(entity, C_i)$ is the updated term frequency for an entity, k is the time window (in hours) set for regular updates, and tf_t is the value of $tf(entity, C_i)$ at time *t*. The parameter k is set to take into consideration the *tf* values of only last k hours and T is the decay factor. Over time there may be a shift in vocabulary of a cluster as more and more tweets enter in it. Sometimes, new clusters represent an updated version of a running story or a terminated cluster - this happens when a story spans a longer period of time than the parameter allows, and can be corrected by tweaking it for individual dataset and/or story characteristics.

Step 5: New Story Detection - A question arises - what if a new story comes up? To preserve the homogeneity of existing clusters and encourage the formation of new ones, its low-frequency entities can be removed by finding the best trade-off point in the curve. This is done using known techniques to find the point on the sorted frequency distribution which is the maximum distance from the line joining the first and the last value [6]. Through our experiments on 5 days' worth of tweets, we found that a cut-off at 95 percentile best served this purpose for this particular dataset. Accordingly, in our experiments, we retained only the top 5 percentile of the entities of a cluster, at the end of every time window.

4. EXPERIMENTAL SETUP

The following paragraphs describe how the SocialStories framework was implemented to identify daily clusters for a month of data.

4.1 Dataset description

The choice of the database was made by considering the top trending topics according to the Twitter dashboard in June 2014; one of the trending topics was "Adobe". By referring to the news, we observed that Adobe announced a major version release on June 17; somewhere in the month, it also announced the findings of its analytical report. It released updates to some products in its suite of creative tools, known as the Creative Cloud. Possibly, Adobe was trending on Twitter because of one or more of the above reasons - however, at the outset it was not clear which story was contributing more or little to this buzz. A literature survey revealed that there were no specific techniques to solve the problem of segmenting the fine-grained stories in an overall topic in social media, so we chose to address this research gap through our experiments. Accordingly, we collected 100% of the 188891 tweets posted under the hashtag "#Adobe" or "Adobe" in the month of June 2014, from Sysomos.com, a commercial social media analytics platform. As seen in Figure 2, the average number of tweets per day was 6290. The highest number of tweets, posted on 18 June, was nearly 35000 tweets.

4.2 Pre-processing

The entity extraction method works on English language tweets only, so first we filtered out non-English tweets, using the NLTK corpus stopwords in Python. Accented characters present in the English tweets were also removed by conversion to unicode. After filtering, the #Adobe dataset comprised 159735 tweets and the trendline of tweets is seen in

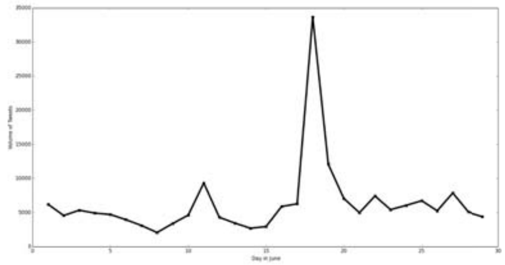

Figure 2: Daily volume of tweets posted about Adobe in June 2014

Figure 2. For the #Adobe dataset, the maximum number of tweets were posted on June 18, the day of the release of the new Creative Cloud. A smaller peak on June 11 marks the release of a new version of FlashPlayer, AIR and AEM. To extract entities - noun phrases, person's and organization's names and hashtags - we used the CMU parser. Given a list of the tweets, the CMU parser identifies the tweets' parts-of-speech and the associated confidences of classificiation. The average distribution of words, entities and hashtags for the #Adobe dataset, as well as the percentage of tweets with at least one hashtag, are provided in Table 1 for reference. The Table justifies the need to boost hashtags, which account for a small proportion of textual contents of tweets but are important contextual indicators.

Number of tweets	**Avg Number of Entities**	**Avg Number of Hashtags**	**Avg number of words**	**Tweets with Hashtags(%)**
159,735	2.74	0.58	15.98	32.32

Table 1: Words, entities and hashtags in the Adobe dataset

4.3 Clustering Parameters

For the incremental clustering algorithm, we input tweets in batches according to a time window; this simulates the nature of streaming tweets in a live system. The size of the time window is data-dependent, to keep the algorithm fast enough to be able to work on huge volumes of streaming data, but accurate to detect homogeneous stories. Typically, window sizes of 1 day would be appropriate for data related to a company, if volume and number of new stories are low on average. During the period of 16-20 June, the hourly volume of tweets about Adobe was high; also, online discussions were about a real offline event - the release of the Creative Cloud. It was suggestive that new stories would break out as more information was released. Keeping these factors in consideration, we set the clustering window size to 1 hour for our experiments. Hashtags were boosted to 1.5 their existing weight. Based on the volume and variety of the tweets coming in, we determined that for the first 200 tweets, the weights would be incremented every 10 tweets, and thereafter, after every hour. Thereafter, the optimal window for decaying the weight of old entities was set to two hours. Using these parameters, the average number of clusters created per day of data was 23. The highest number of clusters were created on June 18, the day with the highest volume of tweets posted.

5. EVALUATION

For evaluation, we have compared the daily stories identified by the SocialStories system for the entire month of June 2014, against hand-curated analytical reports; we have also done a four-way comparison of the stories identified for a five-day period against two baseline Twitter topic modeling systems and the gold standard. First, in the following paragraphs we describe the gold standard and the two baseline approaches.

Hand-curated daily reports - Our gold standard comprised daily social analytics reports about Adobe, hand-curated by professional analysts employed with a marketing analytics company, which identify the emergent stories about Adobe which were being discussed across Twitter and Facebook. For the month of June 2014, there were 31 reported trending Twitter stories, as seen in Table 2. The first column mentions the actual story in the gold standard. For easy reference, they have been assigned a serial number corresponding to the day of the story and the serial number within that day's list of story, for e.g., T23.1 refers to the first story identified on 23rd June. The other columns of the Table provide the number of stories identified by the SocialStories system and the story headlines; these results will be described in Section 6. As seen in the Table, some stories resurface after a few days, such as the Adobe report on TV consumption, which is discussed on June 4 and then again on June 11. In the five day period of our interest, 8 stories are identified in the hand-curated reports, of which T16.1 and T18.3 refer to the same story.

Baseline 1: LDA approach - Latent Dirichlet Allocation is a widely used statistical topic modeling approach by [4] which identifies the top stories in conversation as a mixed proportion of the top stories being discussed. LDA analyzes the words of original texts to discover the themes (as vocabularies that seems to co-occur together) that run through them. It does not require any prior annotation.

Baseline 2: Wavelet approach - The EDCoW algorithm by [16] uses wavelet transformations for event detection in social media. It builds wavelet signals for individual words based on their frequencies, and filters away trivial words by looking at their signal auto-correlations. The remaining words are then clustered to form events with a modularity-based graph partitioning technique.

For comparisons with the baseline approaches, a dataset of all tweets posted between the dates 16-20 June 2014 (both inclusive), was used. This period was chosen because it had the maximum volume of tweets as well as a high number of reported stories in the gold standard - thus it offered a chance to evaluate algorithms on their scalability and precision. For LDA, we have used the Gibbs sampling implementation of LDA [13], which has also been used in the work of [9] and [17]. We pre-processed tweets to remove stopwords, and set the parameters to obtain model 10 stories. For EDCoW, we used the SONDY implementation by [7]. Tweets were tokenised and stopwords removed, then stemming and lemmatization were done, and the tweet stream was divided in windows of size 60 minutes. Based on the experiments by [16], we set the following parameters - *mintermsupport*=0.002; *(thresholdE)*, *maxtermsupport*=1.0, *delta1*=8, *delta2*=48; *gamma*=5. At this juncture, we would like to highlight that although both LDA and EDCoW treat tweets as "bag-of-words", they represent the state of the art in approaches for probabilistic and graph-partitioning approaches - hence they are suitable baselines for our purpose.

5.1 Evaluation Metrics

For evaluation against the gold standard, we have relied on the typical metrics for IR and clustering experiments, which are recommended by [10] for cases where an evaluative benchmark or a gold standard is available. First, we provide a qualitative comparison of the results of the SocialStories system against the gold standard. Next, we measure the coverage, purity, recall and f-scores of the clusters containing 100 tweets or more. For evaluation against the baseline systems, we have used a subset of the data for 16-20 June 2014, when the volume of tweets was the highest, and there were several stories identified in the gold standard report. A qualitative comparison of the stories generated by the two baselines and the SocialStories system is provided.

6. RESULTS

Table 2 provides the results of the SocialStories implementation. Each cluster is considered representative of a single story, and the second column provides the number of distinct automatic clusters, which were considered relevant to a story in the gold standard (provided in the first column). The third column exemplifies the 'story headlines' for some clusters. Considering each cluster as representative of a story, we identified the story headline as the top recurring tweet within the cluster, with a frequency greater than or equal to 30% of the total number of tweets in the cluster. The headline sometimes describes a discussion which has a significantly different focus from what was presented in the overall story, as in the case of story T16.2, when tends to focus more on IBM's superiority in analytics. In cases where multiple relevant stories were identified, each of the headlines seems to indicate a different facet under discussion. This is especially evident for story T4.4, in which case the Camera Raw update is discussed in terms of its availability, its compatibility, its features and its incorporation in Photoshop in the differently themed clusters.

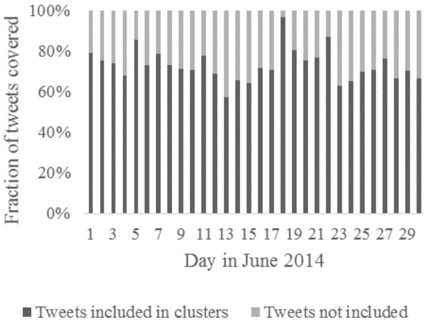

Figure 3: Coverage of the Clustering Algorithm

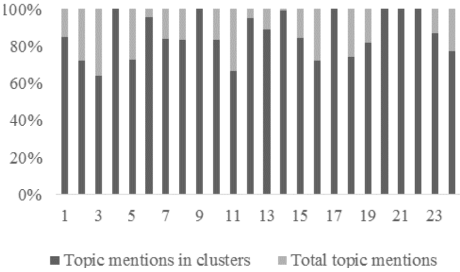

Figure 4: Purity of the clusters

The Table shows that the SocialStories system was able to identify 24 of the 31 reported stories in June 2014. Interestingly, SocialStories was often able to identify the story days in advance of the manual report. For example, both T16.2 and T16.3 were identified on 13th June; T30.1 was identified on 27th June. After compiling our results, we manually searched our dataset for the 7 stories which were not identified by the SocialStories system - although we discovered them in our dataset, we found far fewer number of tweets than those reported in the gold standard. It is possible that the gold standard included tweets which did not explicitly mention Adobe and hence were not captured in our data.The following paragraphs describe the coverage, purity, recall and accuracy of the clusters with more than 100 tweets apiece.

Coverage - Coverage signifies what proportion of all tweets found their way into our selectioned clusters which have more than 100 tweets. Figure 3 illustrates the tweet coverage as a proportion of all the tweets posted in that day - the algorithm is able to retain and cluster close to 80% of all tweets.

Purity - Purity of the clusters measures how many tweets in a cluster, should actually be in the same cluster. Figure 4 describes the purity of the stories - 86% of all tweets relevant to a story are categorized in one of its relevant themes by our algorithm.

Rand Index - The principle behind Rand Index is pairwise recall, and it records the percentage of correct decisions

Table 2. A comparison of actual stories and SocialStories results for June 2014

Actual stories	No. of relevant automatic stories	Example story headlines
T3.1. Adobe Captivate 8 released	1	• New to eLearning? Try Adobe Captivate 8!
T3.2. Microsoft Dynamics marketing and Adobe	1	• Watch Out! Microsoft Released Dynamics Marketing Today - Hey Oracle Salesforce Adobe and other big…
T4.1 Adobe reports on TV consumption	1	• Adobe Report Shows Online TV Consumption at All-Time High Up Nearly 250 ... - Wall Street Journal #techtalk
T4.2 AfterShot 2 Pro competes with LightRoom	1	• Corel AfterShot 2 Pro review: Photo editor a formidable competitor to Adobe Lightroom http://t.co/JYo1p1Z0qT
T4.3 Instagram 2.0 adds PhotoShop-like tools		-
T4.4 Camera Raw 8.5 RC available	5	• T4.4.1 Adobe Camera Raw 8.5 and DNG Converter 8.5 release candidates now available #photography • T4.4.2 Adobe Camera Raw 8.5 Brings In Support for Panasonic Lumix GH4 New Lens Profiles and More.. • T4.4.3 RT @terrylwhite Adobe Camera RAW 8.5 for CC and CS6 Release Candidate is Now Available • T4.4.4 RELEASED: Adobe Camera Raw 8.5 RC (with bug fix for X-T1 and support for TCL-X100) #photography #arts • T4.4.5 Photoshop Camera Raw 8.5 on Adobe Labs - Photoshop Camera Raw 8.5 for CC and Photoshop Camera ...
T4.5 Adobe at MarTech sessions	1	• Check Out the New MarTech Sessions and Speakers from Adobe HubSpot IBM & Marketo a Lowest Rate Expires Frida...
T6.1 Adobe predicts box office losers	1	• Adobe Predicts Summer 2014's Box-Office Losers: The tech company -- boasting a 100 percent success rate for it...
T11.1 Adobe report on TV consumption		-
T11.2 Adobe releases FlashPlayer 14 and AIR 14	1	• Adobe releases Flash 14 and Air 14 with anisotropic filtering Intel x86 Android support and Gamepad API
T11.3 Integrated future of AEM	1	• Adobe Experience Manager: Content Search Social & Mobile a An Integrated Future - http://t.co/hCBokCYpJ9
T11.4 Adobe fixes vulnerabilities in Flash	1	• Adobe updates Flash fixes several vulnerabilities: New Flash Player 6 vulnerabilities many of them critical
T12.1 Chrome PDF Reader goes open		-
T12.2 PhotoShopTouch updates fixes document saving	1	• Adobe Photoshop Touch Updates To Fix Document Saving #tech #gadgets
T12.3 Adobe Digital Index predicts Fifa World Cup 2014	1	• World Cup Most Social Sporting Event Ever Says Adobe: The global social chatter about the 2014 F... #socialmedia
T12.4 Free PhotoShop alternatives		-
T13.1 PhoneGap 3.5.0 Released		-
T16.1 Adobe Q2 earnings preview		-
T16.2 IBM Analytics compared to Adobe metrics	1	IBM: We Have Better CX Analytics than Google or Adobe
T16.3 Adobe ad to run on network TV	1	Adobe Turns to Humor for First Network TV Ad in Over 10 Years
T16.4 The New Creatives Report by Adobe	6	• T16.4.1 RT @TheDrum 36% of creatives still rely on pen and paper for brainstorming @Adobe's New Creatives Report finds • T16.4.2 Adobe survey: Creative professionals thrive … • T16.4.3 RT @Adobe Our latest #NewCreatives report reveals that creative silos don’t exist: http://...
T17.1 Adobe to unlock Creative in Asia	1	• Adobe to 'unlock creative' in Asia with enterprise strategy push
T18.1 Adobe to live stream its CC event	1	• Be there or be square: Adobe will live stream its Creative Cloud keynote address http://t.co/YftxxQgZdL
T18.2 Adonit’s Jot Touch Stylus and CC	1	• Adonit’s latest Jot Touch stylus works with Adobe’s cloud software http://t.co/1vmD6ex8VI

T18.3 Adobe announces Q2 FY14 earnings	1	• Adobe results beat estimates on strong subscription growth: (Reuters) - Adobe Systems Inc the maker of Photos...
T23.1 Judge Koh may decline Adobe settlement	1	• Judge Koh may not approve $324M settlement against Apple Google Intel and Adobe: Employees of major Silicon... http://...
T24.1 Adobe and CMU announce ConstraintJS	1	• Adobe and CMU researchers unveil a brilliant new JavaScript library: ConstraintJS http://t.co/dvzqmRn8QD #venturebeat
T24.2 AYV award 2014 winners announced	1	• Adobe Foundation Announces Winners of the Third Annual Adobe Youth Voices ... - SYS-CON Media (press release):...
T24.3 PhoneGap 3.5.0 released for Android		-
T26.1 Adobe unveils Target Premium	1	• T26.1.1 Adobe Target Premium: It’s Here and It’s the Best Yet
T30.1 Adobe helps in Aperture Migration	5	• T30.1.1 Apple Aperture dies Adobe offers aid to those left behind • T30.1.2 Adobe “fully commited” to helping with Aperture Migration

Table 3. A comparison of actual stories and results from SocialStories, EDCoW and LDA

Actual stories	The SocialStories system	SONDY’s EDCoW implementation	Gibbs’ LDA implementation
T16.2 *IBM Analytics compared to Adobe metrics*	IBM: We Have Better CX Analytics than Google or Adobe http://...		
T16.3 *Adobe AD to run on network TV*	Adobe Turns to Humor for First Network TV Ad in Over 10 Years http://...		
T16.4 *The New Creatives Report by Adobe*	• RT @TheDrum 36% of creatives … • Adobe survey: Creative professionals thrive in a mobile world • RT @Adobe Our latest #NewCreatives report		
T17.1 *Adobe to unlock Creatives in Asia*	Adobe to 'unlock creative' in Asia with enterprise strategy push http://…		
T18.1 *Adobe to live stream its CC event*	Be there or be square: Adobe will live stream its Creative Cloud keynote address	announcement, bad, computer, forward, ill, must, someone, still, watch, yet, you	today, adobe, event, stuff, year, thing, im, launch, tomorrow, stream, youre, lot, whats, subscription, tweet, keynote
T18.2 *Adonit’s Jot Touch Stylus and CC*	Adonit’s latest Jot Touch stylus works with Adobe’s cloud software http://…	ability, app, audition, background, better, cool, cs6, deal, dm, download, even, every, good, icon, illustrator, ink, slide, tell, without, world	adobe, slide, ink, ipad, line, tool, dont, io, mix, pen, anything, ruler, hardware, application, stylus, house, love, word, tablet, creativity
T.18.3 *Adobe announces Q2 FY14 earnings*	Adobe results beat estimates on strong subscription growth		

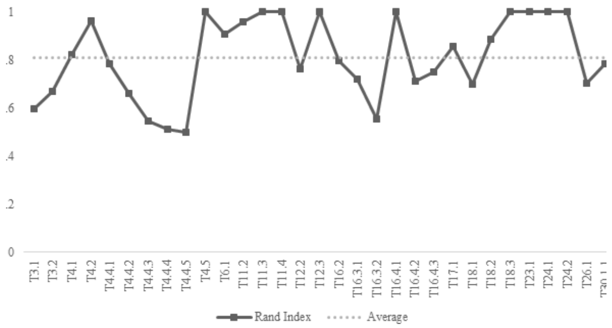

Figure 5: Rand Index for the top tweets in clusters

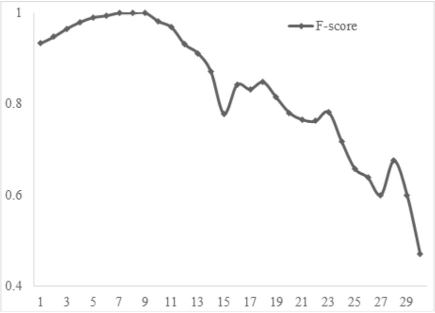

Figure 6: F-scores of the clusters

made by the clustering algorithm. If the most representative tweet of the cluster recurs with a small variation, a good clustering algorithm should assign it into the same cluster as the first time it occurred. By viewing every cluster assignment as a series of decisions, we are able to measure the Rand Index for pairs of the same tweet, reposted during an entire day, through a manual analysis. Figure 4 shows the Rand Index calculated for all the top tweets, with an **average Rand Index of 0.80**. It is seen that the least individual scores were reported for tweets from clusters referring to T4.4, Photoshop's Camera RC update. A deeper analysis identified that although these were the top tweets for the cluster, they were posted less than 20 times; in the case of T4.4.5, 3 of the 4 total posts were assigned to the same cluster; however it has the lowest Rand Index of 0.5. This shows that the Rand Index penalizes tweets with low recurrence. Overall, the Rand Index is high, which reflects that the clustering algorithm performs well as a decision engine.

F-score - Our clustering algorithm succeeded in identifying 25 of the 31 manually-identified stories, with an overall **recall of 80%**. Figure 5 provides the individual F-score values, or the specific accuracy for the clustering of the representative tweets listed in Table 2 into the correct cluster. The F-score is a weighted average of recall and precision. To calculate precision, we took into account false positives, as the irrelevant tweets included in the cluster. To calculate recall, we considered all the false negatives, as the relevant tweets which were not included in the correct cluster. The average F-score is 0.83 with standard deviation 0.14. On comparing with Figure 1, it is evident that the algorithm accuracy is low on days with low tweet volumes, as from day 23 onwards. The system reported several, small, heterogeneous clusters for each of these days, which conveyed no specific story. On these days, there were an average of only 160 tweets posted per hour. Possibly, the hourly window used to cluster tweets and decay entities has worked well on the high volume days before this time period, but was not appropriate for clustering on such days with low hourly volume.

Our results also suggest some insights into which similarity metrics may be working better than others. In days with high tweet volume, giving more weight to inverse cluster frequency and containment may result in more homogeneous clusters; on the other hand, on days with low tweet volume, giving more weight to resemblance may improve the algorithm's performance. Finally, the coefficient of decay in the latter case could be decreased to allow more gradual decay of smaller clusters.

6.1 Comparison against baselines

Table 3 compares the actual stories published during the period of 16-20 June 2014 against the stories identified by SocialStories, and by EDCoW and LDA. First, a comparison against the gold standard reveals that SocialStories performed the best of the three automatic systems, because for the period considered, SocialStories was able to identify 7 of the 8 stories in the gold standard - although T16.1 and T18.3 represent a recurrence of the same story, it was only able to detect the latter occurrence. The output by both, EDCoW and LDA, comprises 'bags of words'; therefore, to interpret and compare their results, we have manually matched their word distribution outputs to the keywords in the stories of the gold standard. EDCoW and LDA systems were only able to identify 2 out of the 7 stories in the gold standard; they thus exhibit low recall. EDCoW's corresponding topic to T18.1 comprises several common words such as "you" and "i'll", which do not contribute any meaning to the topic description. For T18.2, words like "icon" and "illustrator" suggest that two topics may be mixed together.

Next, we present a comparison of the baselines against each other. Because EDCoW and LDA are static algorithms, they generated a total of 12 and 10 stories respectively, for the five-day period; on the other hand, SocialStories follows an hourly, incremental clustering framework, and generated an average of 5 clusters per hour, and 9 clusters reflecting the 7 stories which match the gold standard. This highlights the importance of an incremental, temporal approach for story segmentaton and first story detection; it also suggests that setting finer thresholds for the EDCoW and LDA algorithms may have improved the performance and detection of stories. EDCoW and LDA had 4 stories in common with each other, as compared to only 2 stories in common with the gold standard. The two other matching topics were loosely about Adobe Illustrator and Adobe Muse. The words in topics generated by LDA appear to be more homogeneous, and related to each other, than those generated by EDCoW. Furthermore, LDA appears to have better stop-word filtering than EDCoW.

7. CONCLUSION

The SocialStories system shows promising results in covering, clustering and identifying themes from diverse Twitter data. It overcame the lag in the sequencing and reporting dates of the stories in the hand-curated reports. It is anticipated that with an automatic algorithm such as ours, such delays in reporting important stories could be avoided. The comparisons with the baseline systems highlight some aspects of our approach which give it an edge over the existing state of the art. In SocialStories, every tweet is assigned to a single cluster, which makes it easier to represent each cluster, or story, through its most frequent tweets. In comparison, in LDA and EDCoW, the detected stories with unigram features are difficult for human interpretation; it is not possible to identify which stories may be most relevant or important by looking at the relative sizes of the stories. With EDCoW, the word clustering step could be expensive when the number of bursty words is large; furthermore, the system does not filter out stop-words, which were included in the final output. EDCoW also requires a huge amount of computation, as it uses cross correlation as a similarity measure. SocialStories operates in linear time complexity. The most computationally expensive step, is the pre-processing, because it required the segmentation and filtering of entities from every tweet. Even on June 18, a day with exceedingly high volumes of tweets, our system was able to cluster most of the day's tweets into sensible clusters. Furthermore, it was able to identify the smaller stories reported on this day, which sometimes accounted for only a handful of the thousands of posts for the day. Nevertheless, in these cases we observed that there was often a need for our clusters to be more fine-grained, in case more than one story was assigned to a single cluster. This was often the case for low-volume stories, and especially impacted the precision of the algorithm. In future implementation, we plan to incorporate a parallel computing process to handle streaming tweets on a large scale, and a dynamic time window to adjust for spikes or drops in daily or hourly volumes of topical tweets.

8. REFERENCES

[1] S. Ahmed and M. M. Skoric. My name is khan: The use of twitter in the campaign for 2013 pakistan general election. In *System Sciences (HICSS), 2014 47th Hawaii International Conference on*, pages 2242–2251. IEEE, 2014.

[2] J. Allan. *Topic detection and tracking: event-based information organization*, volume 12. Springer Science & Business Media, 2002.

[3] F. Alvanaki, S. Michel, K. Ramamritham, and G. Weikum. See what's enblogue: real-time emergent topic identification in social media. In *Proceedings of the 15th International Conference on Extending Database Technology*, pages 336–347. ACM, 2012.

[4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[5] M. Cataldi, L. Di Caro, and C. Schifanella. Emerging topic detection on twitter based on temporal and social terms evaluation. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining*, page 4. ACM, 2010.

[6] C. de Mazancourt and U. Dieckmann. Tradeoff geometries and frequencydependent selection. *The American Naturalist*, 164(6):765–778, 2004.

[7] A. Guille. *Diffusion de l'information dans les médias sociaux: modélisation et analyse*. PhD thesis, Université Lumière Lyon 2, 2014.

[8] S. P. Kasiviswanathan, P. Melville, A. Banerjee, and V. Sindhwani. Emerging topic detection using dictionary learning. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 745–754. ACM, 2011.

[9] H. Koga and T. Taniguchi. Developing a user recommendation engine on twitter using estimated latent topics. In *Human-Computer Interaction. Design and Development Approaches*, pages 461–470. Springer, 2011.

[10] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[11] M. Mathioudakis and N. Koudas. Twittermonitor: trend detection over the twitter stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 1155–1158. ACM, 2010.

[12] R. Mehrotra, S. Sanner, W. Buntine, and L. Xie. Improving lda topic models for microblogs via tweet pooling and automatic labeling. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 889–892. ACM, 2013.

[13] X.-H. Phan, L.-M. Nguyen, and S. Horiguchi. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th international conference on World Wide Web*, pages 91–100. ACM, 2008.

[14] H. Sayyadi, M. Hurst, and A. Maykov. Event detection and tracking in social streams. In *ICWSM*, 2009.

[15] C. Shirky. It is not information overload. it is filter failure. 2008.

[16] J. Weng and B.-S. Lee. Event detection in twitter. *ICWSM*, 11:401–408, 2011.

[17] W. X. Zhao, J. Jiang, J. Weng, J. He, E.-P. Lim, H. Yan, and X. Li. Comparing twitter and traditional media using topic models. In *Advances in Information Retrieval*, pages 338–349. Springer, 2011.