Efficient Clustering of Short Messages into General Domains

Oren Tsur **Adi Littman** **Ari Rappoport**
School of Computer Science and Engineering
The Hebrew University
oren@cs.huji.ac.il adi.littman@mail.huji.ac.il www.cs.huji.ac.il/∼arir

Abstract

The ever increasing activity in social networks is mainly manifested by a growing stream of status updating or microblogging. The massive stream of updates emphasizes the need for accurate and efficient clustering of short messages on a large scale. Applying traditional clustering techniques is both inaccurate and inefficient due to sparseness. This paper presents an accurate and efficient algorithm for clustering Twitter tweets. We break the clustering task into two distinctive tasks/stages: (1) batch clustering of user annotated data, and (2) online clustering of a stream of tweets. In the first stage we rely on the habit of 'tagging', common in social media streams (e.g. hashtags), thus the algorithm can bootstrap on the tags for clustering of a large pool of hashtagged tweets. The stable clusters achieved in the first stage lend themselves for online clustering of a stream of (mostly) tagless messages.

We evaluate our results against gold-standard classification and validate the results by employing multiple clustering evaluation measures (information theoretic, paired, F and greedy). We compare our algorithm to a number of other clustering algorithms and various types of feature sets. Results show that the algorithm presented is both accurate and efficient and can be easily used for large scale clustering of sparse messages as the heavy lifting is achieved on a sublinear number of documents.

Introduction

Document clustering is one of the basic tasks required as a first step in many text processing efforts such as topic modeling, content-based recommendation, event detection and discourse analysis, among others. As the popularity of microblogging is growing rapidly, there is thus a need for accurate clustering of micro-messages on a large scale.

While document clustering is well studied, applying traditional clustering methods on micro-messages fails due to the inherent sparseness of micro-messages. Moreover, classic clustering techniques are incapable of handling the massive stream of data posted in microblogging services (e.g. over half a billion messages are posted on Twitter every day). In this paper we propose an accurate and efficient framework for clustering partially tagged micro-messages on services such as Twitter, Google+, Instagram, Pinterest (all support hashtags) as well as news titles or product descriptions. We evaluate our algorithm on a large collection of Twitter tweets.

In the proposed framework the clustering challenge is decomposed into two distinctive tasks: batch clustering of a subset of the data and online clustering of a stream of data. In the batch clustering phase our algorithm exploits tagged data, allowing the conversion of the message (document) clustering task to a tag clustering problem. In order to both overcome sparseness and improve running time, the batch–clustering component works in a bootstrapping manner. First, instead of clustering the documents directly, it creates a collection of virtual non-sparse documents by aggregating all tweets sharing the same hashtag; it then uses a kMeans algorithm (see below) to cluster the hashtags according to the content of the virtual documents. Next, the virtual documents are converted back to short (sparse) documents that are clustered according to the cluster labels assigned to each virtual document in the second stage. Finally, once stable clusters are obtained, the online component can piggy–back on the centroids found in the previous stage in order to assign each new message to a cluster, regardless the length of the message or the fact that it does not contain tags. As the batch component is highly efficient, the batch clustering can be repeated from time to time in order to increase accuracy and capture emerging topics and trends.

This framework is generic in the sense that in the second stage of the batch–clustering component one can use her clustering algorithm of choice. We use kMeans due to its simplicity and its efficiency on the reduced space, compared to the original space. We show that using it in the described manner outperforms other algorithms such as standard kMeans and the mini-batch kMeans that is designed to address scalability and sparseness. In addition, the online component is completely independent and requires only the non-sparse centroids found by the batch component. This decoupling allows one to use less efficient algorithms for the batch clustering given they are more accurate.

Given a fixed number of hashtags, the batch–clustering component scales up in sublinear time as we employ the basic inefficient clusterer only on the reduced space of the hashtags, regardless of the number of tweets or the desired number of clusters. The sublinearity holds under the assumption that the number of hashtags is an order of magnitude smaller than the corpus size. This assumption is very reasonable and holds

in our data. The online component performs in linear time, allowing accurate and efficient clustering of a massive stream of very sparse short messages. We further discuss efficiency in the Clustering Algorithm section.

Although it seems straightforward, a number of steps are required in order to demonstrate the validity of this multistage approach:

1. We first need to show that clustering the virtual documents produces quality clusters of hashtags.
2. We then need to justify the non trivial leap of reducing the tweet clustering challenge to a hashtag clustering task, showing that solving the latter provides a correct solution for the former.
3. Finally, we need to demonstrate that the centroids found in the batch clustering of long and non-sparse virtual documents are capable of accurate clustering of a stream of sparse and untagged messages.

We address all these conceptual steps in our various experimental settings.

This paper is an extension of the short paper by (Tsur, Littman, and Rappoport 2012). While the short paper briefly portrays the use of hashtags for efficient clustering, this paper introduces an online component, effectively clustering tagless tweets. In addition, it provides an exploratory analysis of fine grained clustering into hundreds of clusters as well as a broader discussion of cluster quality in the general domains.

Contribution This work has four main contributions: (i) We provide a novel framework for clustering micro-messages. (ii) We show through extensive evaluations that alternating between dimensions and clustering tasks is applicable to the original task. (iii) Our algorithm is accurate, outperforming other algorithms, and (iv) Our algorithm is fast and scalable thus suitable for large scale tasks such as clustering data from social media streams.

Related Work

While many works focus on document clustering and topic models, clustering of micro-messages is addressed sparsely. The major challenge in clustering of micro-messages (or even short documents) is sparseness, thus traditional techniques in which documents are represented as TF-IDF feature vectors does not perform well.

Banerjee et al. (2007) cluster RSS feed items. They achieve improvement over a baseline expanding the vectors to include key concepts returned by querying Wikipedia with the content of the feed. Kang et al. (2010) use affinity propagation algorithm to cluster similar tweets and Rangrej et al. (2011) conducted a comparative study, comparing three clustering algorithms: kMeans, singular value decomposition and affinity propagation. Experimenting on a small set of tweets they conclude that affinity propagation is best suited for short, though not so sparse, texts[1]. Scalability is not addressed in their comparison.

[1]They avoid sparseness as their collection of only 611 tweets is based on a few search queries.

Sculley (2010) presented a modified kMeans algorithm designed for large scale sparse web page clustering. Our experiments show this modification is not well suited to microblogs (see the Results section).

A growing body of works analyzes Twitter data and Twitter hashtags. Yang and Leskovec (2011) cluster hashtags by temporal patterns of propagation and Romero et al. (2011) manually classify the 500 most popular hashtags, showing that different topical classes present different patterns of stickiness and persistence. Zhao and Jiang (2011) use LDA for fine grained topic modeling of Twitter messages, comparing to topics in New York Times News articles. A fully supervised tag recommender for microblog postings was developed by Garcia et al. (2010), recommending one of 36 hashtags falling under 5 categories.

The discovery of events and emerging stories in Twitter involves clustering of the observed tweets (Petrovic et al. (2010) and Becker et al. (2011)). Both works employ an online clustering technique, however, they do not report results for the clustering stage as the discovered clusters are only one of many heuristic features in the task of event detection. Moreover, sparseness is not an issue in the event detection framework, and cluster quality is neither defined nor evaluated (e.g., by verification against a gold standard). Unlike the specific task of event detection, our work addresses the general problem of clustering sparse short messages to general topical groups, and we provide a comprehensive analysis of cluster quality.

Twitter hashtags can be viewed as topic and sentiment markers. A supervised sentiment classification framework based on data from Twitter is proposed by Davidov et al. (2010), using twitter hashtags as sentiment labels.

Keywords and tags are sometimes used to improve clustering of long documents. Begelman et al. (2006), for example, cluster web pages by co-occurrence of XML tags and Brook and Montanez (2006) compare tag-based clustering of blog posts to standard bag of words algorithms, concluding that tags are too general. All of these works differ from ours in a number of fundamental ways: they cluster long and richly tagged documents thus have no need to address sparseness and secondly, they use tag co-occurrence in order to cluster documents, completely ignoring the document content while we leverage the content in order to overcome sparseness.

Clustering Algorithm

Given a set of tags T, a large set of sparse micro-messages S, where some $d \in S$ contain at least one tag $t \in T$, and given k, the desired number of clusters, we want to find an optimal partition with k sets. Generally, an optimal partition minimizes the within cluster distance and maximize the distance between clusters (we discuss various evaluations methods in the Evaluation Methods Section). In our clustering task we wish to obtain clusters that are based on similarity of content between the documents. The content falls under topical classes, thus we aim at clusters that are similar to a gold standard classification obtained by manual annotation.

Scalable Multi-stage Clustering Algorithm

Although sparse, some domains present texts that are partially tagged. This layer of tags can be exploited in order to overcome sparseness and introduce scalability. The Scalable Multi-stage Clustering algorithm (SMSC) operates in a bootstrapping manner – first it clusters the *tags* according to the full content of the documents, and then uses the clusters of tags and the centroids in order to cluster the *documents* themselves. The "tag" clustering is done in batch mode while the actual document clustering is done in an online manner. The five main stages of the algorithm are detailed next.

Stage 1: creating non-sparse virtual documents Given D a large[2] collection of micro-messages ($D \subset S$) and a set of tags T appearing in D (such that each $d \in D$ contains at least one tag $t \in T$), we create a set of virtual documents D'. The number of virtual documents in D' is $|T|$ – the number of tags in T. Each $d^t \in D'$ is a concatenation of all micro-messages in D that contain a specific tag t. If some d contains more than one tag it will be concatenated to more than one virtual document in D'. Each d^t is now represented as a feature vector based on its words. The vectors based on the virtual documents in D' are not sparse since they are based on the concatenation of multiple micro-messages.

Stage 2: batch clustering of virtual documents We now use a kMeans algorithm in order to cluster the virtual documents in D'. The kMeans algorithm operates on a reduced space of $|T|$, the number of tags, instead of $|D|$. C', the partition achieved for $d^t \in D'$ is actually a solution to a hashtag clustering task, since each $d^t \in D'$ corresponds to a tag $t \in T$. We use $c'_{0<i\leq k}$ to indicate the centroids at the core of the partition C'.

As it is assumed that $|T| << |D|$, the typically time consuming kMeans terminates faster (see Scalability subsection).

Stage 3: assigning tagged sparse documents to clusters In the third stage, each virtual document d^t is redivided to $\{d_i^t\}$ the set of micro-messages it was originally composed from. Each micro-message d_i^t is now being assigned to the same clusters the virtual document d^t was assigned to.

Stage 4: online clustering of sparse documents Given a stream S of short messages, each message s_i is assigned to cluster j according to $\min_j disance(s_i, c_j)$ for $0 < j \leq k$ and any given distance function (we use the same distance function for stages 2 and 4).

If s_i contains a tag $t \in T$, we can avoid the computation of distances and s_i is assigned to the same cluster d^t was assigned to in stage 2.

Stage 5: model adaptation As stages 1 and 2 are highly efficient, they can be repeated from time to time in order to increase accuracy and reflect newly introduced topics, trends and changes in discourse.

Scalability One of the drawbacks of the standard kMeans and other clustering algorithms are their running time in practice. Given N data points and k clusters, and assuming m (m') iterations till convergence[3], the practical running of the kMeans is $T(kMeans) = m \cdot k \cdot N^d + N = O(N^d)$ (d is the dimension of the vector space). On the other hand, in the proposed framework the heavy lifting is done in stage 2 after reducing the number of vectors to cluster in orders of magnitude, thus $T(SMSC) = m' \cdot k \cdot g^d(N) + N$, where g is a sublinear function of N (in our case $g(N) = \sqrt{N}$), thus while asymptotically $T(SMSC) = O(max\{g^d(N), N)\})$, practically, the heavy lifting is done in $O(g^d(N))$ with only two passes on the whole set (N), one pass in the preprocessing (stage 1) and one pass in the reassignment of individual tweets to clusters (stage 3).

Although the batch component (stages 1–3) efficiently clusters a very large set of documents (impractical by explicit batch techniques), it is still incapable of handling a continuous stream of data. Stage 4 clusters incoming documents in an online manner. Once stable centroids are established in the batch phase, each arriving document is now compared only to the k centroids thus performing in a linear time.

Data Representation

We introduce two variations for the representation of the data for the SMSC algorithm.

Bag of words We first took the bag of words approach, representing each $d^t \in D'$ (or $d \in S$) by the *tf-idf* values of the n most frequent words in our data. We experimented with various values of n.

Hashtag co-occurrence Stage 2 of the algorithm could be viewed as a hashtag clustering task. We experiment with a second setting, in which each hashtag in T is represented by a co-occurrence vector, where values are the normalized number of times a pair of hashtags appears in the same tweet. The rationale behind this representation is the assumption that hashtags that belong to the same cluster present higher co-occurrence rate than unrelated hashtags. This type of representation is used by some of the algorithms for clustering long documents such as (Brooks and Montanez 2006). Obviously, co-occurrence representation is only relevant for the batch phase in which only tagged documents are being clustered.

Experimental Framework

The Twitter Data Set

Twitter A Twitter posting is called a *tweet*. A tweet is restricted to 140 characters in length. This length constraint makes characters "expensive". Twitter allows the use of two meta characters: '@' marking a user name (e.g. @BarackObama), and '#' marking a hashtag: a sequence of non whitespace characters preceded by the hash character (e.g. #healthCareReform). The use of hashtags is a popular way for tweeters to provide their readers with some context[4], an

[2] $|D|$ should be big enough to enable an accurate and stable clustering.

[3] We assume m is bounded and relatively small, in our experiments we allow a maximum of 100 iterations, which proved sufficient.

[4] The exact functionality of a hashtag is defined by the practice of the community that uses it. The most frequent uses of hashtags are as a topic marker, as a **bold** typeface intended to draw attention

Category	Definition	size
Music	Referring to songs, albums, groups, movies or TV shows based around music, technology designed for playing music, or events involving any of these. Note that many music groups have unusual names; these still count under the "music" category.	124
Movies	Referring to movies or TV shows, movie or TV studios, events involving a particular movie or TV show, or names of performers who have a movie or TV show specifically based around them. Names of people who have simply appeared on TV or in a movie do not count.	65
Celebrity	Referring to a person or group (e.g. music group) that is featured prominently in entertainment news. Political figures or commentators with a primarily political focus are not included.	77
Technology	Referring to Web sites, applications, devices, or events specifically involving any of these.	98
Games	Referring to computer, video, or twitter-based games, as well as groups devoted to such games.	23
Sports	Referring to sport teams, leagues, athletes, particular sports or sporting events, fan groups devoted to sports, or references to news items specifically involving sports.	46
Idioms	A tag representing a conversational theme on twitter, consisting of a concatenation of at least two common words. The concatenation can't include names of people or places, and the full phrase can't be a proper noun in itself (e.g. a title of a song/movie/organization). Names of days are allowed in the concatenation, because of the Twitter convention of forming hashtags involving names of days (e.g. FollowFriday).	248
Political	A tweet that refers to a politically controversial topic. This can include a political figure, a political commentator, a political party or movement, a group on twitter devoted to discussing a political cause, a location in the world that is the subject of controversial political discussion, or a topic or issue that is the subject of controversial political discussion. Religious comments are considered political due to the debate they raise.	81
None	Every tweet that doesn't fall into one of the 8 categories listed above.	343

Table 1: Definitions of tweet categories (adapted from Romero et al. 2011) and number of hashtags in each category in our corpus.

important function due to the length constraint. For example, the hashtags *#iranelection* and *#freeiran* give the context, topic and ideology behind the tweet "AP: Report: #Iran's paramilitary launches cyber attack http://is.gd/HiCYJU #iranelection #freeiran". Also note the use of *#Iran* as a hashtag and as a crucial (grammatical) part of the sentence.

Hashtags are used extensively and are adopted organically as part of the dynamic communication process. The extensive use of hashtags adds an additional layer of manual "annotation" to a small subset of the data.

Corpus statistics Our corpus consists of over 417,000,000 tweets from Twitter, collected from June 2009 until December 2009. Over three million unique hashtags were observed in our data in over 49 million occurrences, an average of 0.11 hashtag per tweet. The hashtag frequency presents a long tail distribution where the 1000 most frequent hashtags (0.003% of the number of unique hashtags) cover 43% of hashtag occurrences. Hashtags distribution presented in Figure 1. We wish to note that while only a small subset of about 10% of the tweets contain hashtags, they still amount to dozens of millions a day - posing a great challenge from both theoretical and practical perspective. The proposed algorithm not only efficiently cluster this subset – it also accurately clusters the rest of the stream.

Romero et al. (2011) classified the 500 most popular hashtags to nine classes, we followed their classification guidelines, manually classifying the 1000 most popular hashtags in our dataset, creating a gold standard. A thousand tweets were sampled for each of the hashtags in our list, creating a set of one million tweets to cluster.

Following Romero et al. there are nine classes in our gold standard: Music, Movies, Celebrity, Technology, Games, Sports, Idioms, Political and None. The hashtag annotation guidelines are detailed in (Romero, Meeder, and Kleinberg 2011), we slightly alter their guidelines to allow annotation of full tweets. We list the modified guidelines in Table 1

to an important concept and a marker for the agenda of the tweeter.

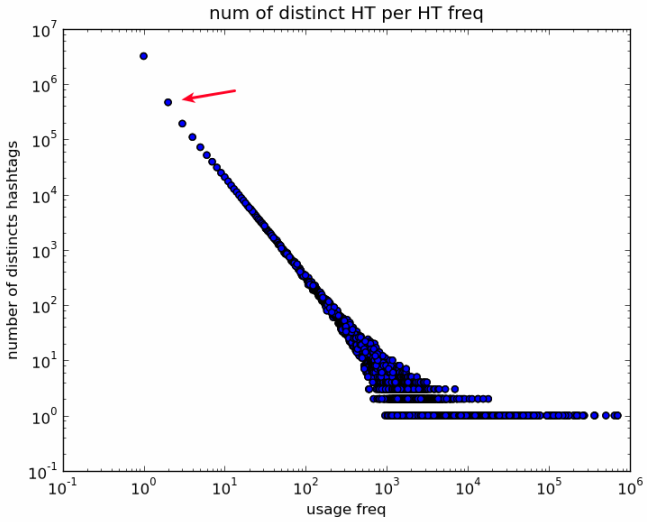

Figure 1: Unique hashtags by number of occurrences. The point marked by the red arrow means that there are about $10^{5.5}$ unique hashtags that appear only twice.

along with the number of hashtags associated with each class. We note that this classification is incomplete as hashtags can potentially have mixed membership, falling under more than one category, e.g *#gagaVmas* (reads as *[Lady] Gaga [in MTv's] Video Music awards*) could be classified under the celebrity, music or movies/TV categories.

Sparseness In this subsection we present some statistics regarding the experimental dataset described above. The experimental dataset contains one million tweets: a thousand tweets for each of the thousand most frequent hashtags. The total number of tokens in this dataset is over twelve millions while the number of unique tokens is slightly over 1.2 million (after decapitalization). Tokens distribution is similar to the

hashtag distribution and matches Zipf's law. The number of words per tweet is normally distributed with an average of 12.2 and a standard deviation of 0.816.

Using all words appearing more than 20 times as features in a vector space, the dimension of the feature space is 27755 and the average number of non-zero entries for each tweet is 8.01. Restricting the dimension to the 1000 most frequent words in the dataset, the average number of non zero entries drops to 5.31.

Number of Clusters

Finding the optimal number of clusters in a kMeans manner is NP-Hard (Mahajan, Nimbhorkar, and Varadarajan 2009). Fixing the dimension of the vector space as k makes the algorithm polynomial (Inaba, Katoh, and Imai 1994). In this work we do not aim at finding the optimal k, instead, we start by following (Romero, Meeder, and Kleinberg 2011), having $k = 9$ (see table 1). However, as noted previously, this proposed classification is incomplete as some tweets can belong to more than one category and some categories could be arguably divided to cleaner sub classes. Moreover, some categories are exceptionally small or exceptionally large, therefore including these categories in our experiment might introduce some bias. In order to accommodate these observations we also report results for $k \in \{6, 7, 8, 9, 10\}$, given by merging, splitting and/or removing classes. Additionally, we present exploratory analysis of clustering of thousands of hashtags into 1000 clusters.

Baseline Algorithms

Three clustering algorithms were used as baseline for comparison: (1) a distribution-based assignment to clusters, (2) standard kMeans, and (3) a modified kMeans designed for web scale sparse data. For both kMeans based baseline algorithms tweets are represented by an n dimensional word vector where, n is the n most frequent words in the corpus and the values are the *tf-idf* scores of these words. We use all words appearing more than 20 times in our data, having $n = 27755$. We also experimented with $n = \{1000, 5000, 10000\}$. We note that the vectors are practically binary and that the vectors are extremely sparse due to the 140-characters restriction of the length of tweets.

Baseline 1 Our first baseline (Dist.BL) assigns tweets to clusters according to the class distribution observed in the gold standard, thus cluster sizes correspond to class sizes. We preferred the distribution of the gold standard over a uniform distribution because it reflects the fact that some topical classes are more popular than others. This choice is more suitable to some of the evaluation measures such as greedy one-to-one, making comparison between the baseline to other algorithms more meaningful.

Baseline 2 We use the classic kMeans (MacQueen 1967), one of the most widely used algorithms for clustering, as our second baseline. Given k, the desired number of clusters, the algorithm randomly chooses k centroids and iterates between two steps: assigning each observation to its closest centroid and updating the centroids to be the mean of each cluster formed in the assignment stage. Thus some variations speed up this process, the standard kMeans algorithm is not well suited for large scale clustering as each iteration computes distances between each data point to each centroid.

Baseline 3 In order to address issues of scalability and sparseness, Sculley (2010) proposed the Web-Scale Fast kMeans (WSFkM) algorithm which includes two modifications of the standard kMeans algorithm: first, he uses mini-batch optimization for kMeans which reduces computation costs by orders of magnitude, and secondly, a projected gradient descent is used, providing a projection into the L1 ball. This algorithm was tested on a set of one million web pages. In this work we test its applicability to much sparser data - a collection of a million tweets.

Evaluation Methods

A meaningful evaluation of clustering results is a challenging task to which many measures have been proposed, see (Pfitzner, Leibbrandt, and Powers 2009; Reichart and Rappoport 2009; Reichart, Abend, and Rappoport 2010) for surveys. Even provided with a gold standard external measure (a correct assignment of elements to classes), there is a number of competing measures that can be used, each has its advantages and disadvantages. In this section we briefly present an array of such measures based on different approaches such as pair-counting, information theory and greedy mapping. Note that measures differ in range and interpretation.

We use the following notation in all of the measures described: Let C indicate the correct classification, let K indicate the induced clusters and a_{ck} the number of items from class c in cluster k.

Rand-Index The Rand Index (Rand 1971) is one of the widely used measures for clusters evaluation.

$$RI = \frac{\sum tp + \sum tn}{\sum tp + \sum fp + \sum tn + \sum fn} \tag{1}$$

Where a true positive pair (tp) is a pair of documents (d_1, d_2) such that $d_1, d_2 \in c_i$ and $d_1, d_2 \in k_j$. True negative (tn) is a pair of documents such that $d_1 \in c_i, d_2 \in c_j, d_1 \in k_i, d_2 \in k_j$ where $i \neq j$. False positive (fp) pair is defined as a pair (d_1, d_2) such that $d_1, d_2 \in k_i$ but $d_1 \in c_i, d_2 \in c_j$ where $i \neq j$. A pair is false negative (fn) if $d_1 \in k_i, d_2 \in k_j, i \neq j$ while $d_1, d_2 \in c_j$. RI is a counting-pairs measure ranging from 0 (worst) to 1 (best).

V-measure The V-measure is an entropy based measure, focused on clusters' homogeneity and completeness (Rosenberg and Hirschberg 2007). Its range is [0,1].

$$V(C, K) = \frac{2hc}{h + c} \tag{2}$$

Where,

$$H(C) = -\sum_{c=1}^{|C|} \frac{\sum_{k=1}^{|K|} a_{ck}}{N} log \frac{\sum_{k=1}^{|K|} a_{ck}}{N} \tag{3}$$

$$H(K) = -\sum_{k=1}^{|K|} \frac{\sum_{c=1}^{|C|} a_{ck}}{N} log \frac{\sum_{c=1}^{|C|} a_{ck}}{N} \tag{4}$$

$$H(C|K) = -\sum_{k=1}^{|K|}\sum_{c=1}^{|C|}\frac{a_{ck}}{N}log\frac{a_{ck}}{\sum_{c=1}^{|C|}a_{ck}} \quad (5)$$

$$H(K|C) = -\sum_{k=1}^{|K|}\sum_{c=1}^{|C|}\frac{a_{ck}}{N}log\frac{a_{ck}}{\sum_{k=1}^{|K|}a_{ck}} \quad (6)$$

$$h = \begin{cases} 1 & H(C,K) = 0 \\ 1 - \frac{H(C|K)}{H(C)} & H(C,K) \neq 0 \end{cases} \quad (7)$$

$$c = \begin{cases} 1 & H(K,C) = 0 \\ 1 - \frac{H(K|C)}{H(K)} & H(K,C) \neq 0 \end{cases} \quad (8)$$

The main disadvantage of the V-measure is that it favors clustering with many small clusters.

VI-measure Like the V-measure, VI-measure (Meila 2007) is also an entropy based measure (VI for Variation of Information).

$$VI(C,K) = H(C|K) + H(K|C) \quad (9)$$

Where $H(C|K)$ and $H(K|C)$ are the same as in the V-measure.

The VI-measure values are between [0, 2log N], where N is the size of the dataset. The disadvantage of this measure is its range. It is hard to compare between results of different datasets in different sizes since the values have different ranges. The VI-measure has two main advantages over the V-measure: it satisfies the metric axioms and it is convexly additive thus changes are local. Splitting or merging clusters only impact the clusters involved.

In contrast to RI and V, the values of the VI-measure decreases as the clustering becomes more complete and more homogeneous. The perfect clustering has $VI = 0$.

NVI-measure NVI-measure (Reichart and Rappoport 2009) is a normalization of the VI-measure.

$$NVI(C,K) = \begin{cases} \frac{H(C|K)+H(K|C)}{H(C)} & H(C) \neq 0 \\ H(K) & H(C) = 0 \end{cases} \quad (10)$$

Where $H(C|K)$, $H(K|C)$, $H(C)$ and $H(K)$ are the same as in the V-measure. This normalization keeps the two advantages of the VI-measure while fixing the range and allowing comparison to other datasets. The values of the NVI-measure decreases as the quality of clusters increases.

Greedy many-to-one This is a simple measure based on mapping clusters to classes in a greedy manner (Zhao and Karypis 2002).

$$GM1(C,K) = \frac{1}{N}\sum_{k=1}^{|K|}\max_{c\in C}\{a_{ck}\} \quad (11)$$

Greedy one-to-one The induced clusters (K) are mapped to the gold standard classes (C) (Reichart, Abend, and Rappoport 2010).

$$G11(C,K) = \frac{1}{N}\sum_{k=1}^{|K|}a_{i_k k} \quad (12)$$

Where i_k is the class index assigned to an induced cluster k by the Kuhn-Munkres algorithm for maximum matching in a complete bipartite graph (Munkres 1957).

F-measure F-measure tailored to multi-class clustering evaluation (Fung, Wang, and Ester 2003):

$$F(C,K) = \sum_{c\in C}\frac{|c|}{N}\max_{k\in K}\{F(c,k)\} \quad (13)$$

$$Recall(c,k) = \frac{a_{ck}}{|c|} \quad (14)$$

$$Precision(c,k) = \frac{a_{ck}}{|k|} \quad (15)$$

$$F(c,k) = \frac{2 * Recall(c,k) * Precision(c,k)}{Recall(c,k) + Precision(c,k)} \quad (16)$$

Gold Standard Sets

All the measures mentioned above require a gold standard of ground truth classification. We computed each of the measures using a number of different gold-standard sets. These annotated sets are shared at the authors web page.

Tweets Gold Standard (TGS) We sampled 1000 tweets from our dataset and had them manually classified by two human annotators. The annotators agreed on 97.4% of the sampled tweets, that constituted the TGS gold standard. In order to classify tweets, our SMSC algorithm reduces the clustering task to a hashtag clustering task. The TGS set allows us to evaluate the algorithm directly on tweets and to verify the conformity between the indirect hashtag-based multi-stage clustering and the content of an individual tweet.

Hashtags Gold Standard (HGS) The core assumption driving this work is that tags can be leveraged for bootstrapping in content based clustering of a sparse collection of documents. In order to classify tweets, our SMSC algorithm reduces the clustering task to a hashtag clustering task. We thus follow the classes definition and classification provided by (Romero, Meeder, and Kleinberg 2011)[5], creating a gold-standard of manually classified *hashtags*. The HGS set allows us to test our algorithm on a larger scale as only the hashtags need to be manually classified.

No Tags TGS (NTTGS) In order to evaluate the online component of our algorithm we use another gold standard set. The NTTGS is the manually annotated set used as the TGS but it is used after the hashtags were removed from the tweets in order to simulate a stream of untagged data.

Results

In this section we report results for the range of experimental settings. All reported results were obtained by averaging experiments in a 10-fold manner. Similar rankings of algorithms were obtained in each individual execution.

[5]Though we used twice as many hashtags.

Alg.	RI ↑	V ↑	VI ↓	NVI ↓	GM1 ↑	G11 ↑	F ↑
Dist.BL	0.68	0.02	3.6	1.95	0.356	0.21	0.24
kMeans	0.72	0.07	3.63	1.96	0.38	0.2	0.22
SMSC	**0.8**	**0.29**	**2.82**	**1.51**	**0.52**	**0.42**	**0.44**

Table 2: Batch clustering results against the tweets gold standard (TGS), vectors based on 10000 most frequent words and random centers. For ease of reading, we indicate (in all tables) whether a higher score is better (↑) or whether a lower score is better (↓).

Alg.	RI ↑	V ↑	VI ↓	NVI ↓	GM1 ↑	G11 ↑	F ↑
Dist.BL	0.72	0.001	3.8	2.01	0.31	0.18	0.21
kMeans	0.75	0.06	3.74	1.96	0.36	0.22	0.23
WSFkM	0.22	0.002	(1.91)	(1.004)	0.31	0.31	0.3
$SMSC_{ht}$	0.78	0.18	3.27	1.72	0.46	0.36	0.42
SMSC	**0.80**	**0.27**	2.94	1.54	**0.5**	**0.4**	**0.46**

Table 3: Batch clustering results against the hashtag gold standard (HGS), vectors based on the 10000 most frequent words and random centers. (See footnote 7 regarding results in parenthesis.)

Batch component

Table 2 presents results of a baseline of the standard kMeans and results of the batch component of our algorithm (SMSC) against the 1000 tweets in the tweet's content gold standard (TGS). Our algorithm's Rand index is 0.8, the Greedy many-to-one is 0.52 and the F score is 0.44. The greedy mapping and the F score values confirm that classification of such a sparse data is hard and that class definitions are fuzzy. However, our algorithm significantly outperforms the baseline in all seven measures (note that in some measures, indicated by ↑, a higher score is better while in other measures, indicated by ↓, a lower score means better results).

Results of the standard kMeans are similar, in most measures, to the distribution-based baseline, demonstrating the challenge posed by sparseness and the limitations of the standard approach. Achieving these results on the TGS set establishes the validity of the reduction from tweets to hashtags that is performed in stage 1 of SMSC.

We next check the scalability of the batch component. Table 3 presents results of four algorithms, evaluated against the million tweets in the HGS set. Our SMSC algorithm outperforms all other algorithms in all measures[6].

Surprisingly, the WSFkM baseline, designed to address large and sparse collection of web pages, performed poorly on our data. Examining the actual clusters produced by the WSFkM baseline, we see that one big cluster was created containing all tweets but a handful. We attribute that to the batch sampling used in order to achieve scalability. This sampling cannot handle extremely sparse micro-messages, though proven suitable for clustering web pages.

[6]The WSFkM baseline seemingly achieves the best results in the VI and NVI measures, however, these results are meaningless as the WSFkM creates one cluster containing almost all instances and $k-1$ clusters each containing 1 or two instances. Since VI and NVI favor big clusters, the case of which most instances are assigned to the same cluster achieves low (good) score. This very peculiarity is discussed in the section *Extreme Cases for the Two Measures* in (Reichart and Rappoport 2009).

Alg.	RI ↑	V ↑	VI ↓	NVI ↓	GM1 ↑	G11 ↑	F ↑
Dist.BL	0.68	0.02	3.6	1.95	0.356	0.21	0.24
kMeans	**0.72**	0.07	3.63	1.96	0.38	0.2	0.22
SMSC	0.71	**0.099**	**3.33**	**1.77**	**0.43**	**0.32**	**0.33**

Table 4: Online clustering results against the tweets gold standard (NTTGS), vectors based on 10000 most frequent words. Centroids are computed by the batch component initialized with random centers. The kMeans results are for standard batch kMeans algorithm.

dim	RI ↑	V ↑	VI ↓	NVI ↓	GM1 ↑	G11 ↑	F ↑
27755	0.79	0.25	2.99	1.56	0.48	0.38	0.43
10000	**0.8**	**0.27**	**2.94**	1.54	**0.5**	0.4	**0.46**
5000	**0.8**	**0.27**	2.96	1.54	0.48	**0.41**	0.44
1000	0.78	0.25	**2.94**	**1.53**	0.48	0.4	0.45

Table 5: Batch clustering results against the HT gold standard for various vector dimension (*d*).

Online component

Table 4 presents results of the online clustering of untagged tweets. The online component of our SMSC algorithm outperforms the baseline in all seven measures and outperforms the standard kMeans in six of the measures while results for the RI are comparable. The results presented in Table 4 can serve only as a basic indication for the quality of clusters achieved by our SMSC, however they do not reflect the real advantage of the SMSC. One should recall that the standard kMeans operates in a batch mode – iterating until a stable partitioning is found, thus impractical for a massive stream of documents, while the online component operates in a linear time with very low constants (the desired number of clusters).

Variations in parameters of the SMSC

In this section we present detailed results for different settings of the SMSC algorithm. In order to further test how SMSC copes with sparseness, we experimented with various dimensions of the vector space defined by the virtual documents. The features for the vectors are the tf-idf values of the n most frequent words in the data, where n varies from 1000 to 27755 (all words appearing more than 20 times in the data). Best results are obtained with $n = 10000$, achieving best score in five of the seven measures, though results on other dimensions are comparable (see Table 5) .

Table 6 lists the average number of non-zero entries in each dimension as well as the running times (in seconds)[7]. While the kMeans clustering takes more than an hour even in the narrowest dimension (top 1000 words) and almost ten hours with the full length vectors (27755 words), the SMSC algorithm converged in 27 seconds in the narrow setting and in only five minutes in the full dimension setting. The Web Scale Fast k-Means baseline converges in less than 2 seconds in all settings, however, we do not report it in Table 6 due to its poor performance on our data (see Table 3). The efficiency of the batch component makes it practical to recompute from

[7]Compare to the average number of non-zero elements in the vectors of the standard kMeans, given in Sparsness subsection.

Alg. / dim	1000	5000	10000	27755
kMeans	5703	6486	18632	34744
SMSC	27	52	129	308
Non-Zero	447.5	1096.7	1438.4	1831.8

Table 6: Convergence time (in seconds) of the batch component, given different dimensions of the vector space. All tests were performed on a single machine with 6 Intel(R) Core(TM) i5-2400 3.10GHz CPU units and 8GB memory. Non-Zero indicates the average number of non-zero elements in the vectors of the virtual documents used for the SMSC.

time to time in order to capture emerging topics and trends, as suggested in stage 5 of the algorithm.

In addition, we used a hashtag co-occurrence based representation ($SMSC_{ht}$ in Table 3) instead of the bag of words representation. While the hashtag co-occurrence based representation performs better than all baselines, it is being outperformed by the bag of words representation based on the virtual documents. This result shows that although related hashtags tend to appear together more than unrelated hashtags, co-occurrence is still relatively sparse, comparing to the bag of words in the virtual documents, thus bootstrapping of content and tags is needed.

Cluster and class distribution analysis

As discussed in the Evaluation Methods section, clustering evaluation measures can be hard to compare and interpret. In this section we provide a deeper view of the clusters and the domain.

In an optimal partition we expect completeness and homogeneity – all tweets of class k will be clustered in a single cluster c, and each cluster c only contains tweets of a single class k. However, this is hardly the case in real world clustering tasks, especially when the data is as sparse, noisy and informal as in social media streams. Twitter idioms, for example, are not expected to be related as what makes a tweet idiomatic is not the real content but the playful addition of an idiomatic hashtag to the content (e.g. '**#theresway2many** ways to procrastinate in the library when you have internet...' vs. '**#theresway2many** microphones being wasted on talentless hacks'). Having more than half of the idioms in one cluster (cluster 5 in Figure 2) suggests that the Twitter 'culture' imposes some patterns on seemingly unrelated idioms. This is a fascinating phenomena from a socio-linguistic perspective, however we leave it for future research.

Another notable class is *games* for which more than 80% of the tweets are clustered in two clusters (clusters 4 & 7), reflecting the fact that the *games* class (as defined in Table 1) has a strong connection to the *technology* class (due to the use of game consoles like X-Box, SPS, WII) and present some confusion with the *sport* class (with terms such as win, lose, game, play etc.).

Tweets of the *music* class present relatively poor completeness and homogenity, as they are scattered between a number of clusters (clusters 2,4 and 9 each contains about 20% of the tweets of this class, the rest is divided between the other five clusters). This is reasonable as many musical tweets carry no real information thus a bag of word approach fails to find similarity. The following four tweets present a somewhat extreme example: (1) "When you love someone and they break your heart, don't give up on love; have faith, restart" #JoeJonas (a fan quoting a song by the Jonas Brothers), (2) PEARL JAM TEN GOLD RECORD AWARD For Sale: http://bit.ly/4szFuW via @addthis #giftideas #christmas and the spam-like[8] (3) @JonasBrothers #BrazilwantsJB #JBcomebacktoBrazil #BrazilwantsJBagain and (4) #justinbieberarmy #justinbieberarmy #teenisland @teenisland #justinbieberarmy #teenisland.

Examination of the homogeneity of clusters (the percentage of each class within the cluster) is also needed due to the high variance in class size (Table 1). For example, while *games*, the smallest class, seems to dominate one of the clusters with 50% of the class instances falling into this cluster (cluster 4, Figure 2), it actually occupies only 10% of the cluster, almost unnoticed among other tweets from much bigger classes (Figure 3).

On the other hand, cluster 5 is dominated by the *idioms* class occupying almost 90% of the cluster and *politics* dominating cluster 8 (Figure 3). Tweets from the class *none* are dominant in all clusters though this is to be expected due to the lack of clear definition of this class and due to the fact that *none* is the biggest class consisting of 34% of our data (see Table 1).

Varying the number of clusters

In order to further learn the connections between classes we experimented with different number of clusters, altering the gold standard accordingly. Some of the variations include merging the *music* and *celebrity* classes (many singers are also teen idols and reality show figures), removing the *none* and/or the *idioms* classes (these classes are unproportionally big and vaguely defined) and splitting *political* to religious and (inter)national politics.

We find that while removing *none* improved results, removing *idioms* actually worsen results. Comparison suggests that while instances of the *none* present low homogeneity and do not share any specific patterns (as this class contains very many topics that were not defined by (Romero, Meeder, and Kleinberg 2011)), a significant part of the *idioms* does share a latent pattern. This is similar to the results observed in the basic $k = 9$ clustering tasks.

Further results for various number clusters and different classes are presented in Table 7. Note that not all the numbers in Table 7 are comparable, as experiments differ in the number N of instances to cluster and in the desired number k of clusters (see Evaluation methods Subection regarding results comparability).

Exploratory analysis of fine grained clustering

In previous sections we presented a clustering framework for large scale clustering of micro-messages into general domains. We provided extensive evaluation, demonstrating that tweets clustering can be reduced to hashtag clustering by concatenation of tweets to longer documents. In this section we briefly discuss the applicability of this framework to finer granularities of clusters. We tried to cluster the 10000 most frequent hashtags into 1000 clusters. We repeated the

[8] Although appearing as noise, these tweets are tweeted and retweeted by fans of the singer Justin Bieber and the Jonas Brothers.

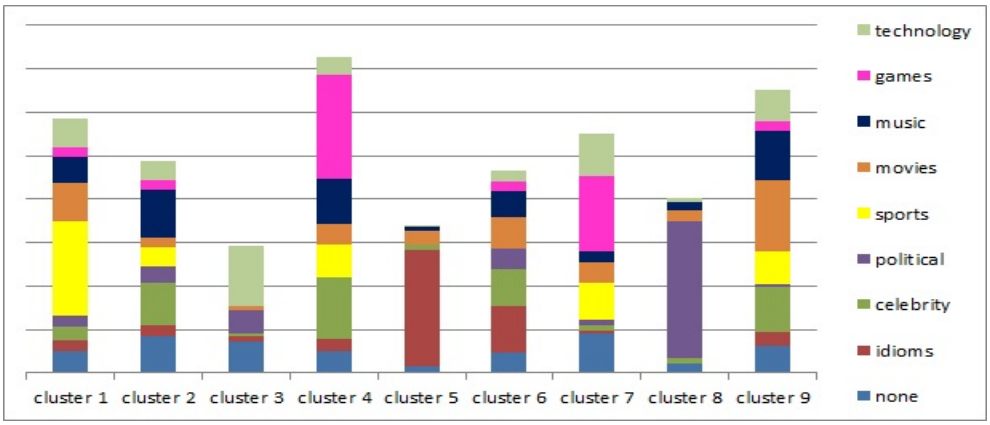

Figure 2: Distribution of gold standard classes in clusters (class completeness). Y axis corresponds to cluster size. The sum of a colored fraction across bars equals to the class size.

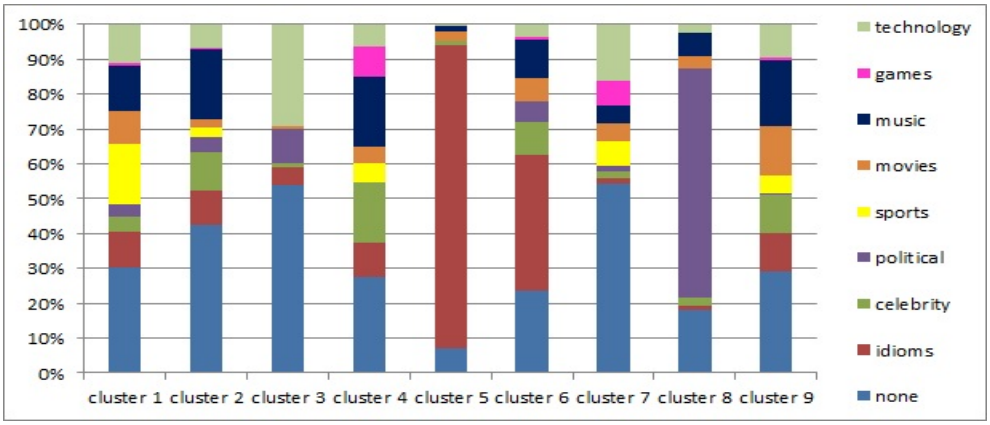

Figure 3: Percentage of class in each cluster with respect to cluster size (cluster homogeneity)

clustering process five times, eventually keeping only the clusters that remained relatively stable. These stable clusters contained 7232 hashtags. Unlike the previous settings described in the sections above, in this setting we do not have an annotated gold standard set that can be used for accurate evaluation of the clustering results. Creating such a gold standard is not feasible as annotators must be well aware of the different nuances of the usage of thousands of hashtags as well as assigning these hashtags to thousand categories. We can thus offer only exploratory evaluation of the results. Exploratory analysis is rather common in the application of clustering techniques in data mining tasks.

The average cluster size is 8.2 with a standard deviation of 15.7, size median is 5 and there are three outliers containing more than 200 seemingly random hashtags. Manually examining the clusters, many seem to be of very good quality, for example the clusters: (#camera #cameras), (#recycle #ecology) and (#neda[9] #freeiran #iranelection #iranrevolution #tehran #iranelecton #mousavi[10] #iranelction #moharram)[11].

One of the interesting findings in this exploratory analysis is that idioms are clustered together in finer grained clusters, e.g. (#justbecause #justcause #justcuz #justbeacuse #justbecuz #justbcuz #jusbecause) although there is no explicitly expected topic for these hashtags. This is in line with the results described in the Cluster and class distribution analysis subection, suggesting that idioms (mainly twitter association games) have a latent structure.

Conclusions

In this work we presented SMSC – a scalable, accurate and efficient multistage clustering algorithm. Our algorithm leverages users' practice of adding tags to some messages by bootstrapping over virtual non sparse documents. Experimenting on a large corpus of tweets from Twitter, we demonstrated that SMSC achieves better results than other clustering algorithms. The algorithm is scalable and more efficient in practice than other algorithms. Moreover, the framework proposed is generic in the sense that the kMeans algorithm

[9]Neda Agha-Soltan – was killed during the 2009 Iranian election protests and became the symbol of the demonstrations.

[10]Mir-Hossein Mousavi – an Iranian politician, leader of the opposition ("green movement") during the 2009 elections.

[11]Typos and variation like #iranelection and #iranelction are as they appear in the data.

Classes	RI ↑	V ↑	VI ↓	NVI ↓	GM1 ↑	G11 ↑	F ↑
base	0.80	0.27	2.94	1.54	0.5	0.4	0.46
(m+c)	0.77	0.22	3.02	1.68	0.49	0.36	0.41
-n	0.82	0.27	2.82	1.5	0.49	0.43	0.49
-i	0.73	0.15	3.2	1.8	0.44	0.29	0.31
-n-i	0.81	0.24	2.84	1.52	0.43	0.38	0.41
-ni(m+c)	0.74	0.19	2.69	1.6	0.5	0.44	0.46
p/2	0.79	0.21	3.3	1.71	0.46	0.32	0.36

Table 7: SMSC clustering results on different numbers of clusters. base: 9 classes detailed in Table 1. (m+c): *music* and *celebrity* merged, -n: *none* removed, -i: *idioms* removed, p/2: *political* is split to 2 classes – *religious* and *national policy*. Evaluation against the HT gold standard, vector size = 10000, and random centers.

used in stage 2 of the algorithm can be replaced by other clustering algorithms, depending on the nature of the desired clusters.

Future research directions include further adaptation of the algorithm to finer granularities. In future research we also plan to address (socio-)linguistic issues such as the latent pattern seemingly exists in the *idioms* class and use the proposed clustering method to further explore other patterns in information diffusion and propagation.

References

Banerjee, S.; Ramanathan, K.; and Gupta, A. 2007. Clustering short texts using wikipedia. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, 787–788. ACM.

Becker, H.; Naaman, M.; and Gravano, L. 2011. Beyond trending topics: Real-world event identification on twitter. In *Fifth International AAAI Conference on Weblogs and Social Media*.

Begelman, G.; Keller, P.; and Smadja, F. 2006. Automated tag clustering: Improving search and exploration in the tag space. In *Collaborative Web Tagging Workshop at WWW2006, Edinburgh, Scotland*, 15–33. Citeseer.

Brooks, C., and Montanez, N. 2006. Improved annotation of the blogosphere via autotagging and hierarchical clustering. In *Proceedings of the 15th international conference on World Wide Web*, 625–632. ACM.

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

Fung, B.; Wang, K.; and Ester, M. 2003. Hierarchical document clustering using frequent itemsets. In *Proceedings of the SIAM International Conference on Data Mining*, volume 30.

Garcia Esparza, S.; O'Mahony, M.; and Smyth, B. 2010. Towards tagging and categorization for micro-blogs. In *21st National Conference on Artificial Intelligence and Cognitive Science, AICS*.

Inaba, M.; Katoh, N.; and Imai, H. 1994. Applications of weighted voronoi diagrams and randomization to variance-based k-clustering. In *Proceedings of the tenth annual symposium on Computational geometry*, 332–339. ACM.

Kang, J. H.; Lerman, K.; and Plangprasopchok, A. 2010. Analyzing microblogs with affinity propagation. In *Proceedings of KDD workshop on Social Media Analytics*.

MacQueen, J. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, 14. California, USA.

Mahajan, M.; Nimbhorkar, P.; and Varadarajan, K. 2009. The planar k-means problem is np-hard. *WALCOM: Algorithms and Computation* 274–285.

Meila, M. 2007. Comparing clusterings–an information based distance. *Journal of Multivariate Analysis* 98(5):873–895.

Munkres, J. 1957. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics* 32–38.

Petrovic, S.; Osborne, M.; and Lavrenko, V. 2010. Streaming first story detection with application to twitter. In *Proceedings of the 11th Annual Conference of the Norht American Chapter of the Association for Computational Linguistics(NAACL HLT 2010*.

Pfitzner, D.; Leibbrandt, R.; and Powers, D. 2009. Characterization and evaluation of similarity measures for pairs of clusterings. *Knowledge and Information Systems* 19(3):361–394.

Rand, W. 1971. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association* 846–850.

Rangrej, A.; Kulkarni, S.; and Tendulkar, A. 2011. Comparative study of clustering techniques for short text documents. In *Proceedings of the 20th international conference companion on World wide web*, 111–112. ACM.

Reichart, R.; Abend, O.; and Rappoport, A. 2010. Type level clustering evaluation: New measures and a pos induction case study. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, 77–87. Association for Computational Linguistics.

Reichart, R., and Rappoport, A. 2009. The nvi clustering evaluation measure. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, 165–173. Association for Computational Linguistics.

Romero, D.; Meeder, B.; and Kleinberg, J. 2011. Differences in the mechanics of information diffusion across topics: Idioms, political hashtags, and complex contagion on twitter. In *Proceedings of the 20th international conference on World wide web*, 695–704. ACM.

Rosenberg, A., and Hirschberg, J. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 410–420.

Sculley, D. 2010. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web*, 1177–1178. ACM.

Tsur, O.; Littman, A.; and Rappoport, A. 2012. Scalable multi stage clustering of tagged micro-messages. In *Proceed-*

ings of the 21th international conference on World wide web. ACM.

Yang, J., and Leskovec, J. 2011. Patterns of temporal variation in online media. In *Proceedings of the fourth ACM international conference on Web search and data mining*, 177–186. ACM.

Zhao, X., and Jiang, J. 2011. An empirical comparison of topics in twitter and traditional media. *Technical Paper Series, Singapore Management University School of Information Systems*.

Zhao, Y., and Karypis, G. 2002. Criterion functions for document clustering: Experiments and analysis.