Temporal Semantics: Time-Varying Hashtag Sense Clustering

Giovanni Stilo and Paola Velardi

Dipartimento di Informatica
Via Salaria, 113 Roma
{Stilo,Velardi}@di.uniroma1.it

Abstract. Hashtags are creative labels used in micro-blogs to characterize the topic of a message/discussion. However, since hashtags are created in a spontaneous and highly dynamic way by users using multiple languages, the same topic can be associated to different hashtags and conversely, the same hashtag may imply different topics in different time spans. Contrary to common words, sense clustering for hashtags is complicated by the fact that no sense catalogues are available, like, e.g. Wikipedia or WordNet and furthermore, hashtag labels are often obscure. In this paper we propose a sense clustering algorithm based on temporal mining. First, hashtag time series are converted into strings of symbols using Symbolic Aggregate ApproXimation (SAX), then, hashtags are clustered based on string similarity and temporal co-occurrence. Evaluation is performed on two reference datasets of semantically tagged hashtags. We also perform a complexity evaluation of our algorithm, since efficiency is a crucial performance factor when processing large-scale data streams, such as Twitter.

1 Introduction

Hashtags are frequently, though not systematically, used by Twitter users to tag the content of their messages. Given the 140 character limits of messages, hashtags provide a natural way to better characterize the topics a message deals with. However, hashtags' popularity surge and decay, and furthermore, the same hashtag might have different meanings in different time periods. For example, recently Jawbone tried a #*knowyourself* campaign on Instagram[1], only to find that the hashtag was already being used generically by thousands of users in all sorts of different contexts.

In addition to polysemy, there is also a problem of synonymy: since new hashtags are freely and continuously introduced by users, different hashtags may share the same meaning, also as a consequence of multilinguality. These two problems reduce the effectiveness of hashtags both as a mean to trace users' interests in time (because of sense shifts), and to capture the worldwide impact of emergent topics (because of synonymy and multilinguality). On the other side, better methods to analyze the semantics of hashtags would be definitely needed, since hashtags are readily available,

[1] http://blog.bufferapp.com/a-scientific-guide-to-hashtags-which-ones-work-when-and-how-many

K. Janowicz et al. (Eds.): EKAW 2014, LNAI 8876, pp. 563–578, 2014.

while textual analysis techniques are limited both by complexity constraints, when applied on large and lengthy micro-blog streams, and by the very reduced dimension of micro-blog texts. Additionally, real-time detection of sense-related hashtags could be used to improve the task of hashtag recommendation, thus further facilitating the monitoring of on-line discussions.

In this paper we propose a methodology for hashtag sense clustering based on temporal co-occurrence and similarity of the related time series. We first convert temporal series into strings of symbols, to reduce complexity. Then, we cluster hashtags co-occurring in the same temporal window and with same, or similar, strings. The paper is organized as follows: in Section 2 we briefly summarize the state of the art on temporal clustering. Section 3 describes our technique to efficiently derive temporal clusters from large and lengthy micro-blog streams. Section 4 is dedicated to performance evaluation. Section 5 analyzes complexity, a relevant issue when dealing with very large data streams. Finally, Section 6 presents our concluding remarks.

2 Related Work

Hashtags have been used in literature to cluster tweets with similar topics. For example, in [1] hashtags are used as a pooling schema to improve LDA topics learned from Twitter. In [2] hashtags are manually associated to a set of 8 categories, plus an additional "catch-all" category. Tweets with hashtags in the same categories are conflated and a model is learned for each category; finally, the model is used for real-time clustering of new messages.

A number of papers deal with hashtag clustering, as we do. The standard technique adopted in literature is based on contextual similarity. In [3] the authors represent a hashtag h by the set of words in the messages including h, and then use K-means on map-reduce to create clusters. In [4] the authors cluster hashtags based on their contextual similarity and then use this information to expand context vectors associated to tweets including these hashtags. In [5] hashtags from different languages are clustered using a machine translation tool, MOSES. Finally, in [6] a combination of co-occurrence frequency, graph clustering and textual similarity is proposed.

As we motivated in the introduction, a better approach seems anchoring hashtag sense clusters to time. A number of works deal with the temporal aspects of hashtags and their persistence: [7] is concerned with the association of usage patterns and hashtag semantics, and [8] analyzes variations in the spread of hashtags. In [9] common shapes of Twitter hashtags are detected using K-Spectral Centroid clustering. Our objective in this paper is however different: rather than using a time-invariant measure of shape similarity to detect "generic" patterns of human attention, we cluster temporally co-occurring hashtags with a similar shape, to induce sense similarity. To the best of our knowledge, this is the first paper in which temporal similarity is used for hashtag sense clustering, however there are a several papers dealing with temporal mining for event detection in micro-blogs [10-17]. Among the most cited, in [10] a temporal analysis technique, named wavelet analysis or EDCoW, is used to discover events in Twitter streams. As a first step, signals are built for individual words by

applying wavelet analysis on the frequency-based raw signals of the words. Autocorrelation is applied to measure the bursty energy of each word. Then, cross-correlation between each pair of bursty words is measured. Finally a cross-correlation table is used to build a graph, and graph-partitioning techniques are applied to discover relevant events. In [11] a technique named TopicSketch is proposed to achieve real-time detection of events in Twitter. Like for EDCoW, events are characterized as "bursty topics", i.e. a set of words showing a sudden surge of popularity followed by a decay. TopicSketch computes in real-time the acceleration (the second order derivative) of three quantities: a) the total Twitter stream; b) each word in the stream; c) each pairs of words in the stream. Given these (known) quantities, the distributions of words over a set of bursty topics $\{T_k\}$ is estimated by modeling the mixture of multiple heterogeneous processes of topics as a Poisson process, and then solving an optimization problem. Hashing techniques and process parallelization are used to keep the problem tractable in terms of memory cost and computational complexity. In fact, one of the main problems with temporal mining when applied to large and lengthy data streams is its computational cost. With respect to these two algorithms, we will show in Section 5 that our method is at least one order of magnitude more efficient.

3 Clustering Hashtag with Symbolic Aggregate ApproXimation (SAX)

In this Section we describe our algorithm, named SAX*, and its application to hashtag sense clustering. The underlying idea of SAX* is that hashtags (or words) with a similar temporal behavior are semantically related. The nature of this relatedness is either limited to a specific temporal slot, e.g. when hashtags describe a unique event (#pope,#habemuspapam), or is more systematic and repetitive, for example when hashtags refer to possibly recurrent, culturally related, issues (such as #followfriday,#thanksgodisfriday). SAX* consists of three steps: in step 1, temporal series of hashtags are sliced into sliding windows and converted into strings of symbols, using Symbolic Aggregate ApproXimation. Then, strings are matched against an automatically learned regular expression representing a generalized pattern of collective attention, in order to discard those hashtags that do not spread across the network. Finally, co-occurring hashtags with similar strings are clustered together.

To tune and evaluate our approach we collected 1% of Twitter traffic, the maximum freely allowed traffic stream, for one year from *June 2012* to *May 2013*, using the standard Twitter API[2]. Other datasets are available, e.g. the Twitter 2011 or 2013 datasets[3], the second being much larger than the first, but still spanning over only two months. A larger time span was indeed necessary to trace a sufficiently large variety of hashtags. Our dataset, hereafter referred to as the *1% Twitter stream*, is about 700 million tweets, with respect to 250 millions tweets of the Twitter 2013 collection, which is, to the best of our knowledge, the largest available collection used so far in micro-blog analysis.

[2] https://dev.twitter.com/docs/streaming-apis
[3] https://sites.google.com/site/microblogtrack/

In what follows, we first describe Symbolic Aggregate ApproXimation (SAX), the algorithm used to represent hashtag time series in a compact way, then, similarly to [9], we identify a class of temporal patterns indicating collective attention. Finally, we present the methodology to obtain SAX* clusters.

3.1 SAX Representation of Time Series

SAX is a technique to reduce a time series of arbitrary length W to a string of arbitrary length N, (typically $N<<W$). Given a time series S(t), this is first normalized through z-score[4] normalization and then discretized, using a well defined dimensionality reduction method called Piecewise Aggregate Approximation [17]. The PAA representation is as follows: given a (normalized) time series $S'(t)$ in a window W, this can be discretized into N partitions of equal length Δ (e.g. days, hours..). We denote with $\bar{s}_i$ (i=1…$N, N = \frac{W}{\Delta}$) the mean value of the function falling into each partition i. Then, the PAA representation is symbolized with a discrete string, using an alphabet Σ: $\{a, b, ..\}$ of n symbols. Since normalized time series have a highly Gaussian distribution, we can determine the breakpoints $\beta_1 .. \beta_{n-1}$ that produce $|\Sigma|$ equally sized areas under the Gaussian curve. Once the breakpoints have been established, the PAA representation is turned into a string of symbols in the following way:

$$\hat{s}_i = j, \qquad j \epsilon \Sigma, \qquad iff\ \beta_{j-1} < \bar{s}_i < \beta_j$$

Figure 1 shows the SAX string (with $\Sigma = 2$ and $\beta = 0$) associated with the normalized time series $S'(t)$ for the hashtag *Olympics*. The series refers to a 10 days window starting on July 25th, 2012, with a 1-day discretization (N=10, Δ= 1 day). The x axis represents the breakpoint and the dashed line shows the $\bar{s}_i$ values. Using the binary alphabet $\{a, b\}$, the correspondent SAX string for *Olympics* is aabbaaaa. Figures 2 and 3 illustrate the effect of z-normalization: Figure 2 shows the time series, in the same window as in Figure 1, for the hashtags: *Olympics, Olimpiadi2012, londra2012, London2012, Londres2012*, while Figure 3 shows the same series after normalization. Even though the five hashtags do not display identical behavior, especially before normalization, their correspondent SAX strings are the same or very similar, intuitively suggesting a correlation.

In our analysis, we are interested only in hashtags whose SAX representation denotes a pattern of collective attention. Rather than clustering time series as in [9], we selected manually a number of words from Wikipedia Events[5] 2011, we generated the SAX strings for these selected words and related hashtags on a previously acquired 1% stream[6], and we used the RPNI algorithm [19], available in the libalf[7] library, to generate compatible regular expressions. With an alphabet of 2 symbols, we finally learned the following regular expression:

---

[4] z-normalization is described in
`http://code.google.com/p/jmotif/wiki/ZNormalization`

[5] `en.wikipedia.org/wiki/Event`

[6] This stream had several holes, therefore we only used to analyze patterns of attention.

[7] `libalf.informatik.rwth-aachen.de/index.php?page=home`

$$(a + b?\, bb?\, a +)?\, (a + b?\, bba *)? \quad (1)$$

This regex captures all the series with one or two peaks and/or plateaus in the analyzed window, such as, for example, the sequences in Figures 1 and 3. Incidentally, we note that this regex turns out to be a generalization of 5 out of 6 shapes of attention learned by the algorithm described in [9][8]. The 6th shape has two major and one minor peak, which would require 3 symbols to be correctly represented.

3.2 SAX* Clustering

As suggested by the example in Figure 3, our aim is to cluster hashtags on the basis of the similarity of their time series. The SAX representation enables this similarity to be captured efficiently.

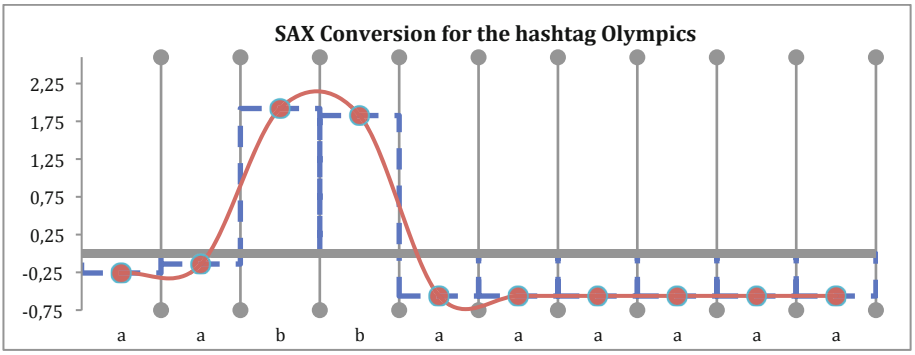

Fig. 1. Binary SAX representation ($|\Sigma| = 2$) of the hashtag Olympics

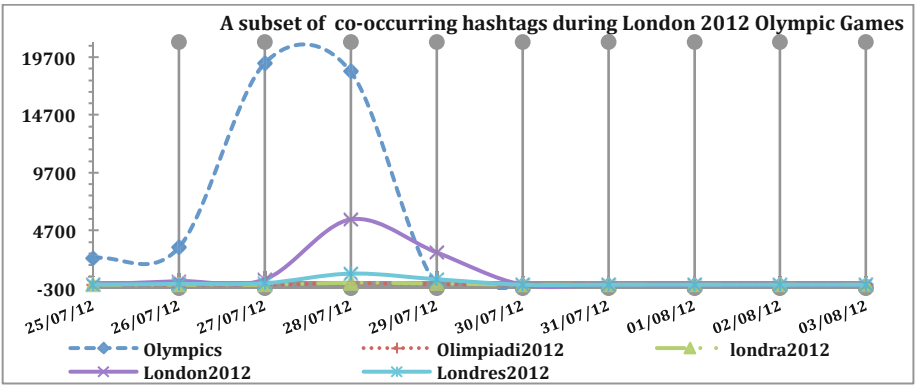

Fig. 2. Non-normalized time series for: Olympics,Olimpiadi2012,londra2012, London2012, Londres2012

[8] See Figure 8 of the mentioned paper, in which 6 shapes of attention of Twitter hashtags are shown.

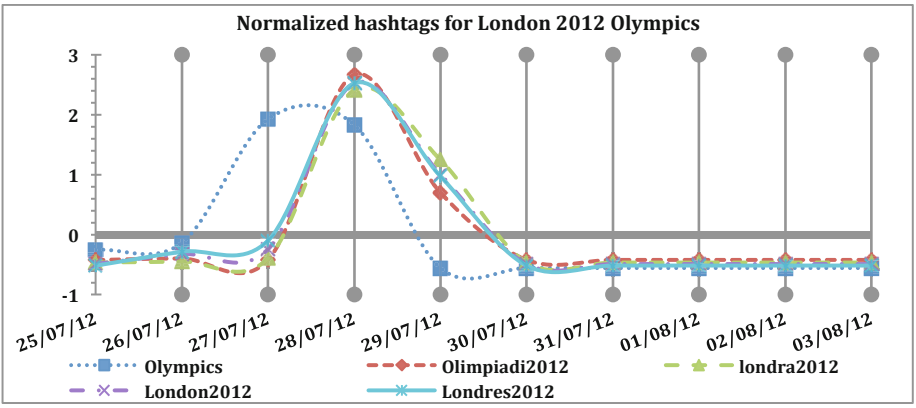

Fig. 3. Normalized time series for: Olympics, Olimpiadi2012, londra2012, London2012, Londres2012

To create clusters, we proceed as follows. In our 1% Twitter stream, we consider sliding windows W_i partitioned in 10 slots of one day each. Our slots are days, but a more fine-grained discretization could be adopted. At each execution of the clustering algorithm, the window is shifted by one day. Within each window, hashtags are converted into SAX binary strings. We also experimented different dimensions for the alphabet Σ (experiments are omitted for the sake of space) eventually finding that best results using our 1% Twitter stream are obtained with a binary alphabet. The parameters $W = 10, \Delta = 1day, |\Sigma| = 2$ perform at best on the 1% stream since this is not sufficiently dense, and only allows it to detect world-wide phenomena. Previous work (e.g. [11]) on a locally dense stream (Singapore tweets during few weeks) has shown that shorter slots (days or hours) should be used to mine geolocated streams.

Given the binary SAX representation of all the hashtags in our Tweet dataset, we consider only those matching the regex (1) in W_i, thereby greatly reducing the computational and memory requirements of the subsequent clustering phase (this will be discussed in more detail in Section 5). Let $L'(W_i)$ be the survived vocabulary of terms in W_i. Over these terms we apply a bottom-up hierarchical clustering algorithm with *complete linkage* [20]. In complete linkage, the algorithm starts with singleton clusters (e.g. each consisting of one term), and then progressively merges two clusters into larger ones, according to the aforementioned "smallest diameter" criterion, measured using a given distance function[9]. We stop hierarchical bottom-up clustering aggregation for a cluster $c_j^{W_i}$ when $SD(d(centroid, t_k)) < \delta$, where SD is the standard deviation of the distance between all terms t_k in $c_j^{W_i}$ and the cluster centroid. We further purge clusters smaller than f elements (f, δ are tunable parameters). Let $C^{Wi} = \{c_1^{Wi}, \ldots c_{ni}^{Wi}\}$ be the clustering result in W_i.

[9] We use the euclidean distance, but other measures, e.g. the edit distance, produce very similar results.

As an example, consider the 10-days window starting on July 27th 2012. In this window, we obtain the following multi-lingual hashtag cluster (the corresponding SAX* sequences are shown only for the cluster centroid):

```
[london2012, londres2012, ItaliansAreSoHot, JJOO, London2012, Londra2012, Londres2012,
Olimpiadi2012, Rai, london2012, londra2012,olimpadi2012, olimpiadi2012, tomorrowland,
London2012, Olympics,olympicceremony,JO2012, JJOO, JO, JO2012, JOLondres2012, Jo2012,
London2012, Londres, Londres2012, Olympics, jo, jo2012, joRTBF]
   [centroid: aaabbaaaaa,  SD: 0.17692605382612614]
```

We can observe that in this case the window perfectly includes the pattern of formula (1), as shown in Figures 2 and 3 (plotting some members of the above cluster), but this is not guaranteed in general. This is the reason for using sliding windows: sliding is better than slicing, since if a slice were to cut a pattern in two we wouldn't be able to detect the correspondent cluster. On the other hand, since consecutive windows overlap over 8 days, we may have many windows and many clusters that capture the same cluster or some of its subtopics. With reference to the previous cluster, the window *W'* obtained by sliding *W* one day to the right, would still generate more or less the same clusters, while sliding three days would miss it, since the correspondent sub-string does not match the regex (1). Therefore, a method is needed to capture the relevant events on a day-to-day basis. To this end, we proceed as follows: For every temporal slot Δ_j (Δ_j is one day in our case) consider the set **W** of all windows such that $\Delta_j \mathrm{C} W_i$ (|**W**|=10 in our case). For each $W_i \in \boldsymbol{W}$, select the subset of clusters $c_{\Delta_j}^{W_i}$ in W_i with a peak in Δ_j, e.g., if a binary alphabet is adopted, whose centroid has a "*b*" in Δ_j. Then, the set of clusters in Δ_j is: $\mathrm{C}^{\Delta_j} = \left\{c_{\Delta_j}^{W_i}\right\}$. Note that clustering is performed on 10-days sliding windows: day clusters are obtained in a post-processing step. Also note that in a day Δ_j there might be zero or more clusters with a peak.

4 Data Analysis and Evaluation

We extracted the hashtags from our 1% Twitter Stream, we removed hastags below a given (language dependent) frequency threshold[10] *f* in each *W* and we then run SAX* with *W*=10, $|\sum| = 2,\ \Delta = 1d, \delta = 0.35$ (see Section 3.1). These parameters were experimentally selected, but given the limited Twitter traffic available (1%), we observed that a more fine-grained analysis (larger alphabet and smaller Δ and *W*) produces less reliable results, as already remarked. Overall, we clustered a set *H* of 124.345 hashtags in 365 sliding windows. The average number of clusters per window was 33.24 (with standard deviation SD=6.76) and the average cluster dimension was 10.29 (SD=3.29). However, except for few examples (as the large *London 2012 Olympic Games* cluster shown in Section 3.2), hashtags are rather cryptic and extensive manual evaluation is almost impossible. To provide an objective evaluation, we designed two experiment: the first is a manual evaluation against two reference

[10] *f* depends on the language and has been set to 99 (English), 17 (Spanish) 4 (French) 3 (Italian), more or less proportional to the relative weight of Tweets in these languages.

classifications, the second provides internal validity measures of cluster cohesion. Both are standard validation approaches adopted in clustering literature and in previous works on hashtag sense clustering [1-5]. Finally, we also evaluate the complexity of SAX* and compare it with other temporal and topic clustering algorithms.

4.1 Experiment 1: Evaluation against Available Datasets

In order to evaluate the quality of extracted clusters, we used two resources:

i) The hashtag classification presented in [2][11]: this dataset (named hereafter TSUR) includes 1000 highly frequent hashtags manually assigned to 9 categories: *Music, Movies, Celebrity, Technology, Games, Sports, Idioms, Political* and *Other*.

ii) A user-populated hashtag taxonomy on the TWUBS on-line hashtag directory[12]: this taxonomy has three top categories (*Event, Organization* and *Topic*) and 32 sub-categories. For example, *Topic* has the following categories: *Art, Education, Entertainement, Gaming, Health&Beauty*, etc. Note that in TWUBS a hashtag may belong to more than one class. We crawled TWUBS and we downloaded about 40,000 hashtags with related classifications.

Both datasets use coarse categories, while our system captures more fine-grained senses, however, as already remarked, a manual evaluation is unfeasible, except for a number of very readable examples like the Olympic game cluster previously shown. The purpose of the evaluation is to show that SAX* clusters are "pure", e.g. most, or all of their members belong the same category. We also note that only a subset of the TSUR and TWUBS hashtags meet the conditions to exceed the threshold f and to match the regex (1) in at least one of the 364 windows W in our 1% Twitter stream. Overall, 243 hashtags from the TSUR dataset and 617 from the TWUBS dataset were also found in our set H of 124.345 hashtags. Let $H(W_i)$be the set of "active" hastags in a window W_i and further let $C^{W_i} = \{c_1 .. c_N\}$[13] be the clustering generated by SAX* in W_i, and $C^T : \{t_1, t_2 .. t_K\}$ the correspondent "ground-truth" classification (either TSUR or TWUBS), such that each cluster t_m includes hashtags belonging to one category[14]. To assess the performance of our system we use the following measure of Precision: $P(C^{W_i}) = \frac{\sum tp_i}{\sum fp_i + \sum tp_i}$ were a true positive pair (tp_i) is a pair of hashtags such that: $h_k, h_j \in c_n in\ C^{Wt}$ $h_k, h_j \in t_m in\ C^T$. Note that we do not use the popular Rand Index[15] since this index takes into account also the false negatives. However with SAX*, two hashtags that do not temporally co-occur are not clustered together, even though they could belong to the same semantic category. For this reason, we

[11] We thank the authors for providing the dataset.

[12] twubs.com/p/hashtag-directory/

[13] We here omit the apex denoting the window in which the cluster is generated, for the sake of simplicity.

[14] Note that K, the number of categories in W_i, is in general lower that the total number of available categories in the two classifications. Also note that TWUBS allows for multiple classifications, therefore some hashtag may belong to more than one t_m.

[15] http://en.wikipedia.org/wiki/Rand_index

cannot compare our results with those in [2]. It is further to be said that the method proposed in that paper, besides being based on contextual similarity rather than temporal similarity, is a trained method while SAX* is *untrained*. In addition to Precision, we also measure the Information Gain[16], defined as the difference between the entropy of the original distribution and that of the derived classification:

$$IG(C^T, C^{W_i}) = \sum_{j=1..K} \left(\frac{|t_k|}{\sum_{j=1..K}|t_J|}\right) log\left(\frac{|t_k|}{\sum_{j=1..K}|t_J|}\right) - \sum_{n=1..N}\left(\frac{|c_n|}{\sum_{j=1..N}|c_J|}\right)\sum_{k=1..K}\frac{|c_n \cap t_k|}{|c_n|} log\left(\frac{|c_n \cap t_k|}{|c_n|}\right)$$

In the formula, with reference to a clustering C^{Wi} of $H(W_i)$, bursty hashtags in window Wi, $K = |C^T|$ is the number of categories of the reference classification (either TSUR or TWUBS) having at least one member in $H(W_i)$, N is the number of clusters generated by SAX* in C^{Wi}, the minuend is the initial entropy of the set $H(W_i)$, i.e. the initial impurity of the examples, and the subtrahend is the weighted sum of entropies of each cluster $c_n \in C^{W_i}$. The IG then provides a measure of the improvement of SAX* over a *baseline classifier* assigning a category based on the a-priori probability distribution of the various categories in $H(W_i)$. We actually compute the normalized IG (NIG), since K may vary in each W_i. Table 1 shows average and standard deviation (SD) of NIG and Precision, over the 365 clusterings C^{Wi} derived in one year.

Table 1. Precision and Information Gain of SAX* in the hashtag clustering task

Golden Classifications:	**TSUR (max K=9)**	**TWUBS (max K=32)**
Average NIG	0.967	0.778
SD(NIG)	0.042	0.1002
Average Precision	0.88	0.77
SD(Precision)	0.127	0.128
Total # of evaluated hashtag pairs	5,678	10,206
Average # of clusters with $\|c_i\|>1$ in W_i	4.85	7.86

The Table shows that the quality of SAX*-induced clusters can be considered indeed very good. The average NIG is close to the maximum of one bit for TSUR and slightly lower for TWUBS, which also has a lower precision. This is coherent with the fact that the number of available categories is more than three times higher for TWUBS (32 against 9) and in addition, in TWUBS some hashtags have multiple classifications. In general, clusters are very pure (e.g. members belong to a unique category), as shown in the following two clustering examples, in which hashtags have been replaced by their semantic labels in TWUBS:

- On: <Jun 01, 2012>: [[MOVIES] [SPORTS,MOVIES,MOVIES] [MOVIES,MOVIES] [SPORTS,SPORTS,SPORTS][SPORTS] [TECHNOLOGY,TECHNOLOGY,GAMES,TECHNOLOGY,GAMES]] (**NIG= 0.920**]

[16] http://en.wikipedia.org/wiki/Information_gain_ratio

- On: <Jun 25, 2012>: [[SPORTS] [MOVIES,MOVIES] [MOVIES] [SPORTS,SPORTS] [IDIOMS,IDIOMS,IDIOMS,IDIOMS,IDIOMS] [POLITICAL,POLITICAL] [MOVIES,MOVIES] [IDIOMS,IDIOMS,IDIOMS,IDIOMS,IDIOMS]] (**NIG=1.00**)

We remark that numbers in Table 1 refer only to the set of hashtags in the two "golden" classifications that also appear in our clusters, since, as stated in the introduction of this Section, our clusters are much larger. As an example, we list some co-clustered pairs with a clear meaning: Giants-sfgiants, MyWeakness-factsaboutme, football-giants, Obama-healthcare, Obama-Obamacare, Dodgers-redsox, apple-iPhone, ff-followfriday, CNN-politics, HabemusPapam-Pope. The examples show that our sense clusters are indeed more fine grained than what captured by the reference classifications, however there is no practically feasible way to evaluate such senses manually. Another problem is that TWUBS and TSUR categories are fixed, and do not capture the temporal shift of hashtag meaning, which was one of the objectives of this paper. Next Section analyzes this issue.

4.2 Experiment 2: Internal Cluster Validity Measures

In this experiment we provide a measure of cluster quality based on the semantic similarity of messages including hashtags in clusters. Similarly to other papers [2,4], we represent each hashtag with a *tfidf* vector of the document D_h^i created by conflating all tweets including a given hashtag h, but we also add the constraint that tweets must co-occur in the same window W_i. We introduce three metrics: the first two are well known measures commonly used to evaluate the similarity of two items D_k^i, D_j^i belonging either to the same or to different clusters in a window W_i. The third-one computes the similarity between the same two items D_i^1, D_j^2 when occurring in two different randomly chosen windows W_{i1}, W_{i2}. The objective of this third measure is to verify our hypothesis of a temporal shift of hashtag meaning. For each hashtag pair $h_k, h_j \in c_n$ and for all clusters $c_n \in C^{Wi}$ detected in window W_i we compute the average intra-clusters similarity $IntraSym(C^{Wi})$, based on the cosine similarity[17] $sym(D_k^t, D_j^t)$:

$$IntraSym(C^{Wi}) = \frac{1}{|C^{Wi}|} \sum_{c_n \in C^{Wi}} \left[\frac{1}{|c_n|(|c_n|+1)} \sum_{\substack{k,h_j \in c_n \\ k \neq j}} sym(D_k^i, D_j^i) \right]$$

Then, for each hashtag pair $h_k \in c_n, h_j \in c_{n'}$ and all clusters C^{Wi} detected in window W_i we compute the average inter-clusters similarity $InterSym(W_i)$ based on the cosine similarity $sym(D_k^i, D_j^i)$:

$$InterSym(C^{Wi}) = \frac{1}{|C^{Wi}|(|C^{Wi}|-1)} \sum_{\substack{c_n, c_{n'} \in C^{Wi}) \\ n \neq n'}} \left[\frac{1}{|c_k||c_{k'}|} \sum_{h_k \in c_n,\ h_j \in c_{n'}} sym(D_k^i, D_j^i) \right]$$

Finally, for each hashtag pair $h_i, h_j \in c_n$ and all clusters C^{Wt} detected in window W_t we compute the average random clusters similarity $RandSym(C^{Wt})$ based on the

[17] *en.wikipedia.org/wiki/**Cosine_similarity***

cosine similarity $sym(D_i^{t_1}, D_j^{t_2})$ when h_i, h_j occur in two non-overlapping randomly selected windows W_{i1}, W_{i2} where i≠ $i_1 \neq i_2$ and $i_1, i_2 \in random(W)$.

$$RandSym(W_i) = \frac{1}{|C^{Wi}|} \sum_{c_n \in C^{Wi}} \left[\frac{1}{|c_n|(|c_n|+1)} \sum_{\substack{h_k, h_j \in c_n \\ k \neq j, i \neq i1 \neq i2 \in random(W)}} sym(D_k^{i_1}, D_j^{i_2}) \right]$$

The main purpose of $RandSym(C^{Wi})$ is to compare the similarity of $D_k^{i_1}, D_j^{i_2}$ when the related tweets co-occur in a cluster in the same window (i= $i_1 = i_2$), and when, instead, they do not co-occur (i≠ $i_1 \neq i_2$). Inspired by the Information Gain, we compute the Similarity Gain by the following formula:

$$SymGain(W_i) = \frac{IntraSym(W_i) - RandSym(W_i)}{RandSym(W_i)}$$

Table 2 shows the values and SD of $IntraSym(W_i), InterSym(W_i)$ and $SymGain(W_i)$.

Table 2. Cluster similarity measures

	Intra	Inter	Rand	Gain
Average	0.2219	0.0083	0.0999	1.3331
St. Deviation	0.2504	0.0042	0.0434	1.6241

The Table shows that, as expected, $IntraSym(W_t) \gg InterSym(W_t)$ but also $IntraSym(W_t) \gg RandSym(W_t)$. This demonstrates the main point of our experiment: hashtag similarity is time-related. Consider for example two hastags, CNN and America, that co-occur in a cluster starting on October 22nd, 2012. Two examples of tweets in this window are (common words are underlined):

#America #CNN
Oct 23rd 2012: *Final presidential debate is tonight tune in #America!!!*
Oct 24th 2012: *Final Debate, Tune in on #CNN*

However, the same two hashtags may be used in very different contexts when found in separate temporal windows, as for example:

#CNN:
Oct 29th, 2012: *Might watch a bit of #CNN to follow #Sandy*
#America:
Dec 14th 2012: *Very sad day in #America. Pray for the families in Connecticut.*

A similar example is provided by the pair Obama, Obamacare:

#Obama,#ObamaCare:
Jun 29th, 2012: *@UserName01 What's your point of view on #OBAMA health care plan?*
Jun 28th, 2012: *#Obamacare Gives millions the opportunity to have health care plan.*

The hashtag Obama however may appear in quite different contexts, such as:

#Obama:
Oct 21st, 2012: *@UserName02 it is in the best interest of #Iran to help President #Obama win. They will say anything to help in the next few weeks*

5 Complexity Analysis

In this Section we perform a complexity evaluation of SAX*, and we compare it with EDCoW [10] and TopicSketch [11]. For SAX*, the complexity analysis is based on [18] and on personal communication with the author; for EDCoW and TopicSketch our computation is based on the algorithm description presented in the respective papers, which we briefly summarize. We introduce the following parameters:

D	number of tweets in W	K	number of discovered events/topics (this is a manually defined parameter in TopicSketch)
t	average document (tweet) length	H	number of hash functions in Topic-Sketch
L	vocabulary dimension (lexicon) in W	I	number of iteration of outer loop in TopicSketch
L'	vocabulary dimension after pruning (when applicable)	i	number of iteration of Newton-Raphson method in TopicSketch
Θ	re-sampling window in EDCoW		
W	window length		

In what follows, in line with [10] and [11], we consider the problem of words temporal clustering rather than hashtags, however the nature of clustered items does not affect the complexity computation.

5.1 SAX* Complexity

The first step requires reading the documents, indexing the terms, and creating a temporal series for every term. Supposing an average length per document of t terms, this step takes *order of* (hereafter the big-o notation will be implicit) Dt. Then, we read the lexicon, pruning all terms below a given frequency, with cost L. Let L' be the pruned lexicon. Finally we remove all terms that do not match the regex (1), with a cost that is linear in the dimension of the window W: $L'W$. Let L'' be the final dimension of the lexicon. The worst case is when $L' = L''$ though in general $L' \gg L''$. The number of comparisons among symbolic strings during hierarchical clustering with complete linkage depends on the string length, which is $\frac{W}{\Delta} = W$ (since $\Delta = 1$) , therefore the worst-case cost is $(L' - 1)(W^2 L')$. After the clustering step, K clusters are generated. Finally, we apply cluster pruning – small clusters are removed - with a cost K. To summarize, the cost is: $Dt + L + L'W + (L' - 1)(W^2 L')$+K

5.2 EDCoW Complexity

A detailed description of the algorithm is found in [10] As for SAX*, the first step consists of the transformation of terms in documents into temporal series with cost Dt. In the first stage D_1 of the algorithm, every term-related signal s_i is converted into another signal s_i'; the new signal is obtained by applying Shannon Wavelet Entropy to sub-sequences of length Θ of the original signal s_i. In other terms a value s_i' is computed every Θ values of s_i. In stage D_2, two contiguous values s_i', s_{i+1}' are aggregated. The cost of the first stage operation is then: $L\left(\frac{W}{\Theta}(\Theta^3+\Theta)\right)$. The second stage filters signals s_i' (of length $\frac{W}{\Theta}$) using the autocorrelation function; this part has a cost $L\left(\frac{W}{\Theta}\log\left(\frac{W}{\Theta}\right)\right)$ and produces a sub-lexicon L'. Next, EDCoW builds the cross-correlation matrix for all pairs of remaining terms. The cost needed to build the cross-correlation matrix is $(L')^2\frac{W^2}{\Theta}$.In the subsequent phase EDCoW detects events though modularity-based graph partitioning that is efficiently computed using *power iteration* at cost L'^2.

For each cluster $e \in$ E (|E|=K) the final cost is bounded by $K\,L'^2$. The final step consists of selecting the clusters on the basis of their related sub-graph and can be included in the previous phase without additional cost. The total cost of the algorithm is then summarized by the following formula:

$$Dt + L\left(\frac{W}{\theta}(\theta^3+\theta)\right) + L\left(\frac{W}{\theta}\,log\left(\frac{W}{\theta}\right)\right) + (L')^2\frac{W^2}{\theta} + K\,L'^2$$

5.3 TopicSketch Complexity

In [11] the authors present a detailed description of the algorithm, though they do not provide a complete complexity analysis. As for the other algorithms, the first step consists of reading the stream and collecting terms statistics with cost Dt. Then a dimension reduction is applied with cost $\mathrm{H}(1 + L\,/\,L')$, where H are hash functions mapping words to bucket $[1\ldots L']$[18] uniformly and independently. The cost of the subsequent phase is summarized by the computational cost of maintaining all the $\mathrm{H}t^2$ accelerations (this cost is provided by the authors). The last step is a topic inference algorithm, modeled as an optimization problem. The gradient-based method[19] to optimize the objective function f is based on the Newton-Raphson approach, whose complexity depends on the multiplication function[20]. Using a very conservative value of 32 bit precision the cost is at least: $I \cdot H \cdot K \cdot i \cdot L' \cdot \log(32)$. Though some minor costs are ignored for the sake of simplicity, the final complexity is order of: $Dt + \mathrm{H}\left(1 + {}^{L}/_{L'}\right) + (\mathrm{H}t^2) + (I \cdot H \cdot K \cdot i \cdot L' \cdot \log(32))$

[18] [1…*B*] in the original paper [11].
[19] Table I of [11].
[20] http://en.wikipedia.org/wiki/Computational_complexity_of_mathematical_operations

5.4 Complexity Estimates

Given the above formulas, we can now provide quantitative complexity estimates. We set the parameters as follows:

- the length *t* of documents is set to 9.4 words[21];
- the size of *D* grows from 100 to 10 million tweets, which is about the actual average size (9.163.437) of English tweets in a 10 days window in a 1% Twitter stream;
- the vocabulary *L* grows according to a Zipfian law with parameter $alfa = 1{,}127$ estimated on our Twitter stream. *L'* grows with the same law (starting from L), with an estimated parameter $alfa = 0{,}41456$.
- $\Theta = 3$ as reported in [10] the window W is 10 days, and Δ=1 day. Note that, in TopicSketch, *W* indirectly impacts on performance, since it limits to a manageable value the dimension *L* of the words to be traced, as the authors say. The impact of W and Δ is accounted by the cost of maintaining the accelerations, Ht^2.
- the number of clusters is set to 50
- according to [11] we set H to 6, I to 50 and *i* to 25.

Table 3 shows that SAX* is one order of magnitude less complex than ECDoW and TopicSketch, on a realistic stream of 10 million tweets. Note that, with respect to the empirical efficiency computation performed in [11], the complexity is here estimated on the theoretical ground and is henceforth independent from parameters, parallelization techniques and computing power. We note that ECDoW is mostly influenced by the first stage of signal transformation and TopicSketch is penalized by the Topic Inference algorithm. Furthermore, while SAX* and ECDoW are not influenced by the K parameter (the number of clusters), using TopicSketch on large Twitter streams with growing K becomes prohibitive, as shown by the complexity formula: in practice, the authors set K=5 in their paper but they do not analyze the effect of this parameter on performance.

Table 3. Complexity analysis as a function of the corpus dimension

D	t	L	L'	Θ	W	K	SAX*	EDCoW	TopicSketch
100	9.4	179	9	3	10	50	7,784	25,341	16,117,823
1K	9.4	2,401	25	3	10	50	73,086	306,630	47,259,589
10K	9.4	32,155	74	3	10	50	665,382	3,820,434	138,620,347
100K	9.4	430,593	217	3	10	50	6,042,708	48,659,378	407,068,448
1M	9.4	5,766,068	635	3	10	50	55,434,549	629,661,338	1,200,080,494
10M	9.4	77,213,473	1,862	3	10	50	**517,658,362**	**8,238,768.,557**	**3,584,819,505**

6 Concluding Remarks

In this paper we introduced a hashtag clustering algorithm based on the novel notion of temporal similarity. We presented SAX*, an algorithm to convert temporal series of hashtags into a sequence of symbols, and then to cluster hashtags with similar and co-occurring sequences. SAX* hashtag clusters, generated from a large and lengthy dataset of Tweets collected during one year, have been evaluated in three ways:

[21] In agreement with `http://firstmonday.org/ojs/index.php/fm/article/view/4366/3654`

- First, we evaluated the quality of clusters using two available datasets of semantically tagged hashtags, showing that SAX* is able to create almost "pure" clusters;
- Second, we used two standard cluster internal validity measures, inter and intra cluster similarity, along with a new measure, the similarity gain. We have shown that tweets including two hashtags h_i, h_j are more similar to each other when they co-occur in the same temporal window and same cluster, than when they occur in different temporal windows;
- Finally, we also conducted a complexity analysis of our algorithm, and compared it with two other temporal clustering methods presented in recent literature, showing that SAX* is one order of magnitude more efficient than the other compared methods.

References

1. Mehrota, R., Sanner, S.: Improving LDA Topic Models for Microblogs via Tweet Pooling and Automatic Labeling. In: SIGIR 2013, Dublin, July 28-August 1 (2013)
2. Tsur, O., Littman, A., Rappoport, A.: Efficient Clustering of Short Messages into General Domains. In: Proceedings of the 7th International AAAI Conference on Weblogs and Social Media, ICWSM 2013 (2013)
3. Muntean, C.I., Morar, G.A., Moldovan, D.: Exploring the meaning behind twitter hashtags through clustering. In: Abramowicz, W., Domingue, J., Węcel, K. (eds.) BIS Workshops 2012. LNBIP, vol. 127, pp. 231–242. Springer, Heidelberg (2012)
4. Ozdikis, O., Senkul, P., Oguztuzun, H.: Semantic Expansion of Hashtags for Enhanced Event Detection in Twitter. In: VLDB 2012 WOSS, Istanbul, Turkey, August 31 (2012)
5. Carter, S., Tsagkias, M., Weerkamp, W.: Twitter hashtags: Joint Translation and Clustering. In: 3rd International Conference on Web Science, WebSci (2011)
6. Modi, A., Tinkerhess, M., Antenucci, D., Handy, G.: Classification of Tweets via clustering of hashtags. EECS 545 Final Project (2011)
7. Posch, L., et al.: Meaning as collective use: predicting semantic hashtag categories on twitter. In: Proceedings of the 22nd International Conference on World Wide Web Companion. International World Wide Web Conferences (2013)
8. Romero, D.M., Meeder, B., Kleinberg, J.: Differences in the mechanics of information diffusion across topics: idioms, political hashtags, and complex contagion on twitter. In: Proceedings of the 20th International Conference Wide Web, ACM (2011)
9. Yang, J., Leskovec, J.: Patterns of temporal variation in online media. In: Proceedings of the fourth ACM International Conference on Web Search and Data Mining, pp. 177–186. ACM (2011)
10. Weng, J., Yao, Y., Leonardi, E., Lee, B.-S.: Event Detection in Twitter. In: ICWSM 2011 International AAAI Conference on Weblogs and Social Media (2011)
11. Xie, W., Zhu, F., Jang, J., Lim, E.-P., Wang, K.: TopicSketch: Real-time Bursty Topic Detection from Twitter. In: IEEE 13th International Conference on Data Mining, ICDM (2013)
12. Qin, Y., Zhang, Y., Zhang, M., Zheng, D.: Feature-Rich Segment-Based News Event Detection on Twitter. In: International Joint Conference on Natural Language Processing (2013)
13. Guzman, J., Poblete, B.: On-line Relevant Anomaly Detection in the Twitter Stream:An Efficient Bursty Keyword Detection Model. In: KDD 2013 (2013)

14. Osborne, M., Petrovic, S., McCreadie, R., Macdonald, C., Ounis, I.: Bieber no more: First Story Detection using Twitter and Wikipedia. In: TAIA 2012 (2012)
15. Diao, Q., Jiang, J., Zhu, F., Lim, E.-P.: Finding Bursty Topics from Microblogs. In: ACL (2012)
16. Naaman, M., Becker, H., Gravano, L.: Hips and Trendy: characterizing emerging trends on Twitter. JASIST (2011)
17. Petrović, S., Osborne, M., Lavrenko, V.: Streaming first story detection with application to Twitter. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT 2010), pp. 181–189. Association for Computational Linguistics, Stroudsburg (2010)
18. Lin, J., Keogh, E., Li, W., Lonardi, S.: Experiencing SAX: A novel symbolic representation of time series. Data Mining and Knowledge Discovery 15(2), 107–144 (2007)
19. Oncina, J., Garcıa, P.: Inferring Regular Languages in Polynomial Updated Time. In: The 4th Spanish Symposium on Pattern Recognition and Image Analysis. MPAI, vol. 1, pp. 49–61. World Scientific (1992)
20. Jain, A.,, K.: Data clustering: 50 years beyond K –means. Pattern Recognition Letters 31, 651–666 (2010)