Semantics-driven Event Clustering in Twitter Feeds

Cedric De Boom
Ghent University – iMinds
Gaston Crommenlaan 8-201,
9050 Ghent, Belgium
cedric.deboom@intec.ugent.be

Steven Van Canneyt
Ghent University – iMinds
Gaston Crommenlaan 8-201,
9050 Ghent, Belgium
steven.vancanneyt@intec.ugent.be

Bart Dhoedt
Ghent University – iMinds
Gaston Crommenlaan 8-201,
9050 Ghent, Belgium
bart.dhoedt@intec.ugent.be

ABSTRACT

Detecting events using social media such as Twitter has many useful applications in real-life situations. Many algorithms which all use different information sources—either textual, temporal, geographic or community features—have been developed to achieve this task. Semantic information is often added at the end of the event detection to classify events into semantic topics. But semantic information can also be used to drive the actual event detection, which is less covered by academic research. We therefore supplemented an existing baseline event clustering algorithm with semantic information about the tweets in order to improve its performance. This paper lays out the details of the semantics-driven event clustering algorithms developed, discusses a novel method to aid in the creation of a ground truth for event detection purposes, and analyses how well the algorithms improve over baseline. We find that assigning semantic information to every individual tweet results in just a worse performance in F_1 measure compared to baseline. If however semantics are assigned on a coarser, hashtag level the improvement over baseline is substantial and significant in both precision and recall.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Experimentation

Keywords

Semantic information, event detection, clustering, social media, Twitter

Published as part of the #Microposts2015 Workshop proceedings, available online as CEUR Vol-1395 (`http://ceur-ws.org/Vol-1395`)

#Microposts2015, May 18th, 2015, Florence, Italy.

1. INTRODUCTION

Traditional media mainly cover large, general events and thereby aim at a vast audience. Events that are only interesting for a minority of people are rarely reported. Next to the traditional mass media, social media such as Twitter and Facebook are a popular source of information as well, but extracting valuable and structured data from these media can be challenging. Posts on Twitter for example have a rather noisy character: written text is mostly in colloquial speech full of spelling errors and creative language use, such posts often reflect personal opinions rather than giving an objective view of the facts, and a single tweet is too short to grasp all the properties that represent an event. Nevertheless the user-contributed content on social media is extensive, and leveraging this content to detect events can complement the news coverage by traditional media, address more selective or local audiences and improve the results of search engines.

In the past researchers mostly used textual features as their main source of information to perform event detection tasks in social media posts. Next to the text itself, other characteristic features such as the timestamp of the post, user behavioural patterns and geolocation have been successfully taken into account [1, 4, 15, 17, 18, 22]. Less used are so-called semantic features, in which higher-level categories or semantic topics are captured for every tweet and used as input for the clustering algorithm. These semantic topics can either be very specific—such as sports, politics, disasters. . . —or can be latent abstract categories not known beforehand; such an abstract topic is usually a collection of semantically related words. In most applications semantics are determined on event level after the actual event detection process [19]. We however propose to use semantic information on tweet level to drive the event detection algorithm. After all, events belonging to different semantic categories—and thus also its associated tweets—are likely to be discerned more easily than semantically related events. For example then it is relatively easy to distinguish the tweets of a sports game and a concurrent politics debate.

The use case we address in this paper consists of dividing a collection of tweets into separate events. In this collection every tweet belongs to a certain event and it is our task to cluster all tweets in such a way that the underlying event structure is reflected through these clusters of tweets. For this purpose we adopt a single pass clustering mechanism. As a baseline we use a clustering approach which closely resembles the algorithm proposed by Becker et al. to cluster Flickr photo collections into events [2, 3], and in which we only use plain textual features. We then augment this baseline algorithm, now incorporating semantic information about the tweets as a second feature next to the text of the tweet. As it turns out, solely using a semantic topic per tweet only marginally improves baseline performance; the attribution of semantic labels on tweet level seems to be too fine-grained to be of any predictive value. We therefore employ an online dynamic algorithm to assign semantic topics on hashtag level instead of tweet level, which results in a courser attribution of topic labels. As will be shown in this paper, the latter approach turns out to be significantly better than baseline performance.

The remainder of this paper is structured as follows. In Section 2 we shortly discuss the most appropriate related work in recent literature, after which we describe the methodology to extract events from a collection of Twitter posts in Section 3. The collection of data and the construction of a ground truth is treated in Section 4. Finally we analyse the results of the developed algorithms in Section 5.

2. RELATED WORK

Since the emergence of large-scale social networks such as Twitter and their growing user base, the detection of events using social information has attracted the attention of the scientific community. In a first category of techniques, Twitter posts are clustered using similarity measures. These can be either based on textual, temporal, geographical or other features. Becker et al. were among the first to implement this idea by clustering a Flickr photo collection [2, 3]. They developed a single pass unsupervised clustering mechanism in which every cluster represented a single event. Their approach however scaled exponentially in the number of detected events, leading to Reuter et al. improving their algorithm by using a prior candidate retrieval step [15], thereby reducing the execution time to linear scaling. Petrović et al. used a different technique based on Locality Sensitive Hashing, which can also be seen as a clustering mechanism [14]. In this work, tweets are clustered into buckets by means of a hashing function. Related tweets are more probable to fall into the same bucket, which allows for a rapid comparison between tweets to drive the event detection process.

The techniques in a second category of event detection algorithms mainly use temporal and volumetric information about the tweets being sent. Yin et al. for example use a peak detection strategy in the volume of tweets to detect fire outbreaks [22], and Nichols et al. detect volume spikes to identify events in sporting games [13]. By analysing communication patterns between Twitter users, such as peaks in original tweets, retweets and replies, Chierichetti et al. were able to extract the major events from a World Cup football game or the Academy Awards ceremony [7]. Sakaki et al. regarded tweets as individual sensor points to detect earthquakes in Japan [17]. They used a temporal model to detect spikes in tweet volume to identify individual events, after which a spatial tracking model, such as a Kalman filter or a particle filter, was applied to follow the earthquake events as they advanced through the country. Bursts of words in time or in geographic location can also be calculated by using signal processing techniques, e.g. a wavelet transformation. Such a technique was successfully used by Weng et al. in their EDCoW algorithm to detect Twitter events [21], and by Chen and Roy to detect events in Flickr photo collections on a geographic scale [6].

Semantic information is often extracted after the events are detected to classify them into high level categories [16]. This can be done in either a supervised way, using a classifier like Naive Bayes or a Support Vector Machine, but most of the times unsupervised methods are preferred, since they do not require labelled data to train models and are able to discover semantic categories without having to specify these categories beforehand. Popular unsupervised techniques are Latent Dirichlet Allocation (LDA), clustering, Principal Component Analysis (PCA) or a neural auto-encoder. LDA was introduced by Blei et al. in 2003 as a generative model to extract latent topics from a large collection of documents [5]. Since then many variants of LDA have emerged tailored to specific contexts. Zhao et al. created the TwitterLDA algorithm to extract topics from microposts, such as tweets, assuming a tweet can only have one topic. Using community information next to purely textual information, Liu et al. developed their own version of LDA as well, called Topic-LinkLDA [10]. A temporal version of LDA, called TM-LDA, was developed by Wang et al. to be able to extract topics from text streams, such as a Twitter feed [20]. By batch grouping tweets in hashtag pools, Mehrotra et al. were able to improve standard LDA topic assignments to individual tweets [12].

3. EVENT CLUSTERING

In this section we will describe the mechanics to discover events in a collection of tweets. In the dataset we use, every tweet t is assigned a set of event labels E_t. This set contains more than one event label if the tweet belongs to multiple events. The dataset itself consists of a training set T_{train} and a test set T_{test}. The details on the construction of the dataset are found in Section 4. We will now try to recover the events in the test set by adopting a clustering approach. First the mechanisms of an existing baseline algorithm will be expounded. Next we will extend this algorithm using semantic information calculated from the tweets.

3.1 Baseline: Single Pass Clustering

Our baseline algorithm will use single pass clustering to extract events from the dataset. Becker et al. elaborated such an algorithm to identify events in Flickr photo collections [2, 3]; their approach was criticized and improved by Reuter et al. for the algorithm to function on larger datasets [15]. In this paper we will adopt single-pass clustering as a baseline that closely resembles the algorithm used by Becker et al.

As a preprocessing step, every tweet in the dataset is represented by a plain tf-idf vector and sorted based on its timestamp value. In the following we will use the same symbol t for the tweet itself and for its tf-idf vector. As the algorithm proceeds, it will create clusters of tweets, which are the retrieved events. We denote the cluster to which tweet t belongs as S_t; this cluster is also characterized by a cluster center point s_t. We refer to a general cluster and corresponding cluster center point as resp. S and s. The set A contains all clusters which are currently active, i.e. being considered in the clustering procedure. During execution of the algorithm, a cluster is added to A if it is newly created. After some time a cluster can become inactive by removing this cluster from the set A. In Section 5 we will specify how a cluster can become inactive.

The baseline algorithm works as follows. When the current tweet t is processed, the cosine similarity $\cos(t, s)$ between t and cluster center s is calculated for all S in A. A candidate cluster S'_t (with cluster center s'_t) to which t could be added, and the corresponding cosine similarity $\cos(t, s'_t)$, are then calculated as

$$S'_t = \arg\max_{S \in A} \cos(t, s), \tag{1}$$

$$\cos(t, s'_t) = \max_{S \in A} \cos(t, s). \tag{2}$$

If S'_t does not exist—this occurs when A is empty—we assign t to a new empty cluster S_t, we set $s_t = t$ and S_t is added to A. If S'_t does exist, we need to decide whether t belongs to this candidate cluster or not. For this purpose we train a logistic regression classifier from LIBLINEAR [8] with a binary output. It takes $\cos(s'_t, t)$ as a single feature and decides whether t belongs to S'_t. If it does, then we set S_t to S'_t and we update its cluster center s_t as follows:

$$s_t = \frac{\sum_{t \in S_t} t}{|S_t|}. \tag{3}$$

If t does not belong to S'_t according to the classifier, then as before we assign t to a new empty cluster S_t and we set $s_t = t$.

In the train routine we assign every tweet one by one to a cluster corresponding to their event label. At every step we calculate the candidate cluster S'_t for every tweet t in T_{train} and verify whether this cluster corresponds to one of the event labels of t in the ground truth. If it does, we have a positive train example, otherwise a negative example. The number of positive and negative examples are balanced by randomly removing examples from either the positive or negative set, after which the examples are used to train the classifier.

In the original implementation by Becker et al. the processing of a tweet is far from efficient since every event cluster has to be tested. After a certain time period, the amount of clusters becomes very large. The adjustments by Reuter et al. chiefly aim at improving this efficiency issue. We do not consider these improvements here, since in Equation (1) we only test currently active clusters, which is already a performance gain.

3.2 Semantics-driven Clustering

To improve the baseline single pass clustering algorithm we propose a clustering algorithm driven by the semantics of the tweets. For example tweets that belong to the same semantic topic—e.g. sports, disasters, ...—are more likely to belong to the same event than tweets about different topics. Discerning two events can become easier as well if the two events belong to different categories.

To calculate a semantic topic for each of the tweets in the dataset, we make use of the TwitterLDA algorithm [23]. It is an adjustment of the original LDA (Latent Dirichlet Allocation) algorithm [5] for short documents such as tweets, in which every tweet only gets assigned a single topic—instead of a probabilistic distribution over all the topics—and single user topic models are taken into account. After running the TwitterLDA algorithm, every tweet t gets assigned a semantic topic γ_t.

The actual clustering algorithm has the same structure as the baseline algorithm, but it uses the semantic topic of the tweets as an extra semantic feature during clustering. We define the semantic fraction $\sigma(t, S)$ between a tweet and an event cluster as the fraction of tweets in S that have the same semantic topic as t:

$$\sigma(t, S) = \frac{|\{t' : t' \in S \wedge \gamma_{t'} = \gamma_t\}|}{|S|}. \tag{4}$$

To select a candidate cluster S'_t (with cluster center s'_t) to which t can be added, we use the cosine similarity, as before, as well as this semantic fraction:

$$S'_t = \arg\max_{S \in A} \cos(t, s) \cdot \sigma(t, S). \tag{5}$$

We choose to multiply cosine similarity and semantic fraction to select a candidate cluster since both have to be as large as possible, and if one of the two factors provides serious evidence against the candidate cluster, we want this to be reflected. Now we use both $\cos(t, s'_t)$ and $\sigma(t, S'_t)$ features to train a logistic regression classifier with a binary output. The rest of the algorithm continues in the way the baseline algorithm does.

3.3 Hashtag-level Semantics

As pointed out by Mehrotra et al. the quality of topic models on Twitter data can be improved by assigning topics to tweets on hashtag level instead of on tweet level [12]. To further improve the semantics-driven clustering, we therefore use a semantic majority voting scheme on hashtag level, which differs from the approach by Mehrotra et al. in that it can be used in an online fashion and that we consider multiple semantic topics per tweet.

In the training set we assign the same topic to all tweets sharing the same event label by performing a majority vote:

$$\forall t \in T_{\text{train}} : \gamma_t = \arg\max_{\gamma} \left|\{t' : \gamma_{t'} = \gamma \wedge E_{t'} \cap E_t \neq \emptyset\}\right|. \tag{6}$$

This way every tweet in the training set is represented by a semantic topic that is dominated on the level of the events instead of on tweet level, resulting in a much coarser attribution of semantic labels. We cannot do this for the test set, since we do not know the event labels for the test set while executing the algorithm. We can however try to emulate such a majority voting at runtime. For this purpose, every tweet t is associated with a set of semantic topics Γ_t. We initialize this set as follows:

$$\forall t \in T_{\text{test}} : \Gamma_t = \{\gamma_t\}. \tag{7}$$

Next to a set of topics for every tweet, we consider a dedicated hashtag pool H_h for every hashtag h, by analogy with

[12]. With every pool H we associate a single semantic topic β_H. As the algorithm proceeds, more and more hashtag pools will be created and filled with tweets.

When a tweet t is processed in the clustering algorithm, it will first be added to some hashtag pools, depending on the number of hashtags in t. So for every hashtag h in t, t is added to H_h. When a tweet t is added to a hashtag pool H, a majority vote inside this pool is performed:

$$\beta_{\text{new},H} = \arg\max_{\gamma} \left|\{t' : t' \in H \wedge \gamma_{t'} = \gamma\}\right|. \tag{8}$$

We then update Γ_t for every tweet t in H:

$$\forall t \in H : \Gamma_{\text{new},t} = (\Gamma_{\text{old},t} \setminus \{\beta_H\}) \cup \{\beta_{\text{new},H}\}. \tag{9}$$

Finally $\beta_{\text{new},H}$ becomes the new semantic topic of H. Note that every tweet t keeps its original semantic topic γ_t.

What still needs adjustment in order for the clustering algorithm to use this new information, is the definition of the semantic fraction from Equation (4). We altered the definition as follows:

$$\sigma'(t, S) = \max_{g \in \Gamma_t} \frac{|\{t' : t' \in S \wedge g \in \Gamma_{t'}\}|}{|S|}. \tag{10}$$

Since Equation (10) implies Equation (4) if Γ_t contains only one element for every tweet t, this is a justifiable generalization.

4. DATA COLLECTION AND PROCESSING

In the past many datasets have been assembled to perform event clustering on social media. Unfortunately many of these datasets are not publicly available; this is especially true for Twitter datasets. We therefore choose to build our own dataset, available at `http://users.ugent.be/~cdboom/events/dataset.txt`. To speed up this task we follow a semi-manual approach, in which we first collect candidate events based on a hashtag clustering procedure, after which we manually verify which of these correspond to real-world events.

4.1 Event Definition

To identify events in a dataset consisting of thousands of tweets, we state the following event definition, which consists of three assumptions. Assumption 1 – a real-world event is characterized by one or multiple hashtags. For example, tweets on the past FIFA world cup football matches were often accompanied by hashtags such as #USAvsBelgium and #WorldCup. Assumption 2 – the timespan of an event cannot transgress the boundaries of a day. This means that if a certain real-world event takes place at several days—such as a music festival—this real-world event will be represented by multiple event labels. The assumption will allow us to discern events that share the same hashtag, but occur on a different day of the week, and will speed up the eventual event detection process. The hashtag #GoT for example will spike in volume whenever a new episode of Game of Thrones is aired, which are thus different events according to our definition. Assumption 3 – there is only one event that corresponds to a certain hashtag on a given day.

Assumption 3 is not restrictive and can easily be relaxed. For example if we would relax this Assumption and allow multiple events with the same hashtags to happen on the same day, we would need a feature in the event detection process to incorporate time differences, which is easily done. Alternatively we could represent our tweets using df-idf$_\text{t}$ vectors, instead of tf-idf vectors, which also consider time aspects of the tweets [1].

4.2 Collecting Data

We assembled a dataset by querying the Twitter Streaming API for two weeks, between September 29 and October 13 of the year 2014. We used a geolocation query and required that the tweets originated from within the Flanders region in Belgium, at least by approximation. Since only very few tweets are geotagged, our dataset was far from a representative sample of the tweets sent during this fortnight.

We therefore augment our dataset to make it more representative for an event detection task. If a real-world event is represented by one or more hashtags (Assumption 1), then we assume that at least one tweet with these hashtags is geotagged and that these hashtags are therefore already present in the original dataset. We thus consider every hashtag in the original dataset and use them one by one to query the Twitter REST API.

A query to the REST API returns an ordered batch of tweets $(t_i)_{i=1}^m$, where m is at most 100. By adjusting the query parameters—e.g. the maximum ID of the tweets—one can use multiple requests to gather tweets up to one week in the past. To make sure we only gather tweets from within Flanders, the tokens in the user location text field of every tweet in the current batch are compared to a list of regions, cities, towns and villages in Flanders, assembled using Wikipedia and manually adjusted for multilingual support. If the user location field is empty, the tweet is not considered further. We define a batch $(t_i)_{i=1}^m$ to be valid if and only if

$$\frac{|\{t_i : t_i \text{ in Flanders}\}|}{\text{timestamp}(t_m) - \text{timestamp}(t_1)} > \tau_1, \tag{11}$$

where τ_1 is a predefined threshold. If there are τ_2 subsequent invalid batches, all batches for the current considered hashtag are discarded. If there are τ_3 batches in total for which less than τ_4 tweets were sent in Flanders, all batches for the current considered hashtag are discarded as well. If none of these rules apply, all batches for the current hashtag are added to the dataset. When the timestamp$(\cdot)$ function is expressed in minutes, we set $\tau_1 = 1$, $\tau_2 = 12$, $\tau_3 = 25$ and $\tau_4 = 10$, as this yielded a good trade-off between execution time and quality of the data.

4.3 Collecting Events

Using the assembled data and the event definition of Section 4.1 we can assemble a ground truth for event detection in three steps. Since events are represented by one or more hashtags according to Assumption 1, we first cluster the hashtags in the tweets using a co-occurrence measure. Next we determine whether such a cluster represents an event, and finally we label the tweets corresponding with this cluster with an appropriate event label.

To assemble frequently co-occurring hashtags into clusters, a so-called co-occurrence matrix is constructed. It is a three-dimensional matrix Q that holds information on how many times two hashtags co-occur in a tweet. Since events can only take place on one day (Assumption 2), we calculate co-occurrence on a daily basis. If hashtag k and hashtag ℓ co-occur $a_{k,\ell,d}$ times on day d, then

$$\forall k, \ell, d\colon Q_{k,\ell,d} = \frac{a_{k,\ell,d}}{\sum_i a_{k,i,d}}. \tag{12}$$

To cluster co-occurring hashtags we adopt the standard DBSCAN clustering algorithm. This is an online clustering algorithm that requires two thresholds to be set: the minimum number of hashtags $\min_h$ per cluster and a minimum similarity measure ϵ between two hashtags above which the two hashtags reside in the same ϵ-neighbourhood. The similarity measure between hashtags k and ℓ on day d is defined as

$$\mathrm{sim}_{k,\ell,d} = \frac{Q_{k,\ell,d} + Q_{\ell,k,d}}{2}. \tag{13}$$

If we run DBSCAN for every day in the dataset, we obtain a collection of clusters of sufficiently co-occurring hashtags on the same day.

A lot of these clusters however do not represent a real-world event. Hashtags such as #love or #followme do not exhibit event-specific characteristics, such as an isolated, statistically significant peak in tweet volume per minute, but can rather be seen as near-constant noise in the Twitter feed. In order to identify the hashtags that do represent events and to filter out the noise, we follow a peak detection strategy. For this purpose we treat each cluster of hashtags separately, and we refer to the hashtags in these clusters as 'event hashtags'. With each cluster C we associate all the tweets that were sent on the same day and that contain one or more of the event hashtags in this cluster. We gather them in a set T_C. After sorting the tweets in T_C according to their timestamp, we calculate how many tweets are sent in every timeslot of five minutes, which makes up for a sequence $(v_{C,i})_{i=1}^{n}$ of tweet volumes, with n the number of time slots. We define that some v_{C,i^*} is an isolated peak in the sequence $(v_{C,i})$ if and only if

$$v_{C,i^*} \geq \theta_1 \wedge \forall i \neq i^*\colon v_{C,i^*} \geq v_{C,i} + \theta_2, \tag{14}$$

with θ_1 and θ_2 predefined thresholds. Only if one such isolated peak exists (Assumption 3), we label all tweets t in T_C with the same unique event label e_t and add them to the ground truth. Since we used the event hashtags from C to construct this event, we have to remove all event hashtags in C from the tweets in T_C, otherwise the tweets themselves would already reflect the nature of the events in the ground truth.

With this procedure it is however likely that some tweets will belong to multiple events, but only get one event label. This is possible if a tweet contains multiple event hashtags that belong to different event hashtag clusters. We therefore alter the ground truth in which every tweet t corresponding to an event is associated with a set of event labels E_t instead of only one label. Of course, for the majority of these tweets, this set will only contain one event label.

In our final implementation we set $\min_h = 1$, $\epsilon = 0.3$, $\theta_1 = 10$ and $\theta_2 = 5$. These values were chosen empirically, such that, with these parameters, clusters of co-occurring hashtags are rarely bigger than three elements. After manual inspection and filtering, the final dataset contains 322 different events adding up to a total of 63,067 tweets. We assign 2/3 of the events to a training set and 1/3 to a test set, leading to 29,844 tweets in the training set and 33,223 in the test set.

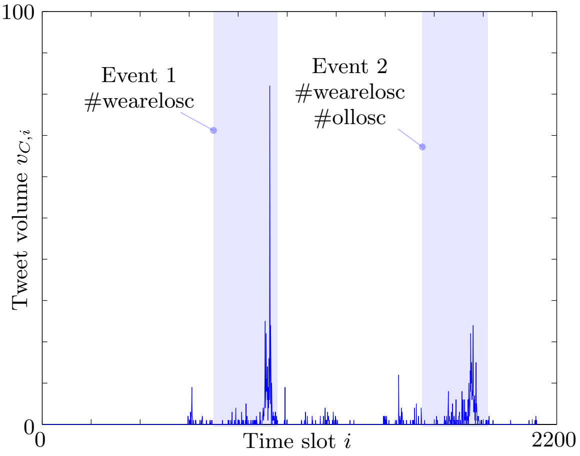

Figure 1: Plot of tweet volume in function of time slot for two example events in the dataset, with their associated hashtags.

Figure 1 shows a plot of the tweet volume in function of time slot for two events in the dataset. The plot only covers the first week in the dataset. The events are two football games of the French team LOSC Lille—which is a city very near Flanders, and therefore shows up in our dataset. The first event is characterised by the single hashtag #wearelosc, and the second event by two hashtags: #wearelosc and #ollosc. Our algorithm detects the peaks in tweet volume during the games, and since only one significant peak exists per day, we assign the same event label to all tweets with the associated hashtags sent during that day.

The final dataset is made available at the earlier mentioned URL. We provide for every tweet its tweet ID, timestamp, corresponding event labels and event hashtags, and whether it belongs to either the training or test set. Due to Twitter's restrictions, we cannot directly provide the text of all tweets.

5. RESULTS

5.1 Performance Measures

To assess the performance of the clustering algorithms, we report our results in terms of precision P, recall R and F_1 measure, as defined in [3, 15], and restated here:

$$P = \frac{1}{|T|} \sum_{t \in T} \frac{|S_t \cap \{t' \colon e_{t'} = e_t\}|}{|S_t|}, \tag{15}$$

$$R = \frac{1}{|T|} \sum_{t \in T} \frac{|S_t \cap \{t' \colon e_{t'} = e_t\}|}{|\{t' \colon e_{t'} = e_t\}|}, \tag{16}$$

$$F_1 = 2 \cdot \frac{P \cdot R}{P + R}, \tag{17}$$

in which T stands for the total dataset of tweets. When tweets can have multiple event labels, these definitions however do not apply any more. We therefore alter them as

	Precision	Recall	F_1-measure
Baseline	47.12%	35.35%	40.40%
Semantics-driven	52.80%	30.60%	38.74%
Hashtag semantics	48.62%	36.97%	42.00%
Baseline (multi)	64.96%	36.36%	46.62%
Semantics-driven (multi)	69.27%	31.47%	43.28%
Hashtag semantics (multi)	64.06%	37.77%	47.52%

Table 1: Using hashtag-level semantics clearly outperforms baseline and plain semantics-driven clustering.

	Purity	Number of events
Baseline	61.29%	409
Semantics-driven	64.76%	662
Hashtag semantics	61.15%	441
Baseline (multi)	75.51%	409
Semantics-driven (multi)	77.74%	662
Hashtag semantics (multi)	73.72%	441

Table 2: A comparison of baseline, plain semantics-driven clustering and hashtag semantics in terms of purity and number of event clusters.

follows:

$$P = \frac{1}{|T|} \sum_{t \in T} \max_e \frac{|S_t \cap \{t' : e \in E_{t'} \wedge e \in E_t\}|}{|S_t|}, \qquad (18)$$

$$R = \frac{1}{|T|} \sum_{t \in T} \max_e \frac{|S_t \cap \{t' : e \in E_{t'} \wedge e \in E_t\}|}{|\{t' : e \in E_{t'} \wedge e \in E_t\}|}. \qquad (19)$$

Note that Equations (18) and (19) imply Equations (15) and (16) if there is only one event label per tweet.

We will also use purity as an indicator of the quality of the event clusters we obtain. We have chosen the definition of purity as in [11] and adapted it to our context as follows:

$$\text{purity} = \frac{1}{|T|} \sum_{t \in T} \max_e \frac{|S_t \cap \{t' : e = e_{t'}\}|}{|S_t|}. \qquad (20)$$

It is a measure that is closely related to precision. For multiple event labels, we alter this measure to the following expression:

$$\text{purity} = \frac{1}{|T|} \sum_{t \in T} \max_e \frac{|S_t \cap \{t' : e \in E_{t'}\}|}{|S_t|}. \qquad (21)$$

5.2 Results

We now discuss the results of the algorithms explained in Section 3 with the use of the dataset constructed in Section 4. In the algorithms we make use of a set A of active event clusters, which become inactive after some time period. We could for example use an exponential decay function to model the time after which a cluster becomes inactive since the last tweet was added. Using Assumption 2 however we can use a much simpler method: when a new day begins, all event clusters are removed from A and thus become inactive. This way we start with an empty set A of active clusters every midnight.

For the semantics-driven clustering algorithm we assign the tweets to 10 TwitterLDA topics using the standard parameters proposed in [23] and 500 iterations of Gibbs sampling. Table 1 shows the results of the baseline algorithm, the semantics-driven algorithm and the hashtag-level semantics approach, both for one event label and multiple event labels per tweet. Note that, since we have removed the event hashtags from the tweets in the ground truth, the hashtag-level semantics approach does not use any implicit or explicit information about the nature of the events.

We note that the hashtag-level semantics approach outperforms the baseline clustering algorithm, with an increase of 1.6 percentage points in F_1-measure for single event labels. In terms of precision and recall, hashtag-level semantics performs better in both metrics than baseline in the single label case (significant improvement, $p < 0.001$ in t-test). When using multiple event labels per tweet, precision is decreased by 0.9 percentage points, but raises recall with 1.4 percentage points, leading to an increase of F_1-measure by 0.9 percentage points.

Compared to the standard semantics-driven algorithm we do 6 percentage points better in recall, but 4 percentage point worse in precision for single event labels. Hashtag-level semantic clustering seems to manage to account for the substantial loss in recall that occurs when using the basic semantics-driven method, but lacks in precision; the precision is however still 1.5 percentage points better than the baseline algorithm. The plain semantics-driven approach is 1.7 percentage points worse than baseline in terms of F_1-measure, but provides much more precision by sacrificing in recall. For multiple event labels the differences are even more pronounced between the standard semantics approach and the other algorithms. The former performs 3.3 percentage points worse in F_1-measure compared to baseline, and 4.2 percentage points worse compared to hashtag semantics. Using multiple event labels, the plain semantics-driven algorithm however has a much higher precision than baseline and hashtag semantics.

To assess the significance of the differences in F_1 measure between our three systems, we used a Bayesian technique suggested by Goutte et al. [9]. First we estimated the true positive, false positive and false negative numbers for the three systems. Next we sampled 10,000 gamma variates from the proposed distribution for F_1 for these systems and calculated the probability of one system being better than another system. We repeated this process 10,000 times. Hashtag semantics resulted in a higher F_1 measure in 99.99% of the cases; our results are thus a significant improvement over baseline. By contrast, the plain semantics-driven approach is significantly worse than baseline, also in 99.99% of the cases. Concerning multiple event labels, the hashtag semantics approach is better in 98.5% of the cases than baseline, which is also a significant improvement—although less than in the single event label case.

We also compare our three approaches in terms of cluster purity and the number of detected event clusters. These numbers are shown in Table 2. We see that the purity of the clusters in the plain semantics-driven approach is higher than baseline and hashtag semantics, but the number of detected event clusters is even substantially larger. This explains the high precision and low recall of the semantics-driven algorithm. The purity of baseline and hashtag semantics is almost equal, but the latter approach discerns more events than baseline, thereby explaining the slight increase in precision and recall for the hashtag semantics approach compared to baseline. Concerning multiple event labels, the purity increases significantly compared to single event labels. Since the number of detected events remains the same, this explains the substantial increase in precision for the multi-label procedure.

5.3 An Illustrative Example

As a matter of example, consider the tweet *"we are ready #belgianreddevils via @sporza"*. This tweet was sent on the occasion of a football game between Belgium and Andorra—the Belgian players are called Red Devils and the airing television channel was Sporza. Since most tweets on this football game were sent in Dutch or French, the baseline clustering approach is not able to put this tweet in the correct cluster, but rather in a cluster in which most tweets are in English. This tweet is however related to a sports-specific topic, so that in both the semantics approaches the tweet is assigned to a correct cluster. It is clear that the hashtag #belgianreddevils has something to do with sports—and in particular a football game of the Belgian national team—but there exist tweets that contain this hashtag and that have not been categorized into the sports category by the TwitterLDA algorithm. For example the tweet *"met 11 man staan verdedigen, geweldig! #belgiumreddevils"* (which translates to "defending with 11 men, fantastic!") belongs to a more general category. This shows that calculating semantic topics on tweet level results in a fine-grained, but also more noisy assignment of these topics, which is reflected in the number of detected events shown in Table 2. By assigning the semantic topics on hashtag level however, all tweets with the hashtag #belgianreddevils will eventually belong to the sports category. It will result in a coarser, less detailed assignment of the topics, resulting in a more accurate event detection, and fewer detected events.

6. CONCLUSION

We developed two semantics-based extensions to the single-pass baseline clustering algorithm as used by Becker et al. to detect events in Twitter streams. In this we used semantic information about the tweets to drive the event detection. For this purpose we assigned a topic label to every tweet using the TwitterLDA algorithm. To evaluate the performance of the algorithms we semi-automatically developed a ground truth using a hashtag clustering and peak detection strategy, to aid the manual labelling of tweets with events. When using the topic labels at the level of individual tweets, the algorithm performs significantly worse than baseline. When however gathering the semantic labels of the tweets on a coarser, hashtag level we get a significant gain over baseline. We can conclude that high-level semantic information can indeed improve new and existing event detection and clustering algorithms.

7. ACKNOWLEDGMENTS

Cedric De Boom is funded by a Ph.D. grant of Ghent University, Special Research Fund (BOF).
Steven Van Canneyt is funded by a Ph.D. grant of the Agency for Innovation by Science and Technology in Flanders (IWT).

8. REFERENCES

[1] L. M. Aiello, G. Petkos, C. Martin, D. Corney, S. Papadopoulos, R. Skraba, A. Goker, I. Kompatsiaris, and A. Jaimes. Sensing Trending Topics in Twitter. *Multimedia, IEEE Transactions on*, 2013.

[2] H. Becker, M. Naaman, and L. Gravano. Event Identification in Social Media. In *WebDB 2009: Twelfth International Workshop on the Web and Databases*, 2009.

[3] H. Becker, M. Naaman, and L. Gravano. Learning similarity metrics for event identification in social media. In *WSDM '10: Third ACM international conference on Web search and data mining*, 2010.

[4] H. Becker, M. Naaman, and L. Gravano. Beyond Trending Topics: Real-World Event Identification on Twitter. In *ICWSM 2011: International AAAI Conference on Weblogs and Social Media*, 2011.

[5] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Machine Learning*, 2003.

[6] L. Chen and A. Roy. Event detection from flickr data through wavelet-based spatial analysis. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, 2009.

[7] F. Chierichetti, J. Kleinberg, R. Kumar, M. Mahdian, and S. Pandey. Event Detection via Communication Pattern Analysis. In *ICWSM '14: International Conference on Weblogs and Social Media*, 2014.

[8] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A Library for Large Linear Classification. *The Journal of Machine Learning Research*, 2008.

[9] C. Goutte and E. Gaussier. A probabilistic interpretation of precision, recall and F-score, with implication for evaluation. In *ECIR'05: Proceedings of the 27th European conference on Advances in Information Retrieval Research*, 2005.

[10] Y. Liu, A. Niculescu-Mizil, and W. Gryc. Topic-link LDA: joint models of topic and author community. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, 2009.

[11] C. Manning, P. Raghavan, and H. Schütze. *An Introduction to Information Retrieval.* Cambridge University Press, 2009.

[12] R. Mehrotra, S. Sanner, W. Buntine, and L. Xie. Improving lda topic models for microblogs via tweet pooling and automatic labeling. 2013.

[13] J. Nichols, J. Mahmud, and C. Drews. Summarizing sporting events using twitter. In *IUI '12: Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*, 2012.

[14] S. Petrović, M. Osborne, and V. Lavrenko. Streaming first story detection with application to Twitter. In *HLT '10: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2010.

[15] T. Reuter and P. Cimiano. Event-based classification of social media streams. In *ICMR '12: Proceedings of the 2nd ACM International Conference on Multimedia Retrieval*, 2012.

[16] A. Ritter, Mausam, O. Etzioni, and S. Clark. Open domain event extraction from twitter. In *KDD '12:*

Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, 2012.

[17] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes Twitter users: real-time event detection by social sensors. In *WWW '10: Proceedings of the 19th international conference on World wide web*, 2010.

[18] G. Stilo and P. Velardi. Time Makes Sense: Event Discovery in Twitter Using Temporal Similarity. In *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2014 IEEE/WIC/ACM International Joint Conferences on*, 2014.

[19] S. Van Canneyt, S. Schockaert, and B. Dhoedt. Estimating the Semantic Type of Events Using Location Features from Flickr. In *SIGSPATIAL '14*, 2014.

[20] Y. Wang, E. Agichtein, and M. Benzi. TM-LDA: efficient online modeling of latent topic transitions in social media. In *KDD '12: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012.

[21] J. Weng, Y. Yao, E. Leonardi, and B.-S. Lee. Event Detection in Twitter. In *ICWSM '11: International Conference on Weblogs and Social Media*, 2011.

[22] J. Yin, A. Lampert, M. Cameron, B. Robinson, and R. Power. Using social media to enhance emergency situation awareness. *IEEE Intelligent Systems*, 2012.

[23] W. X. Zhao, J. Jiang, J. He, Y. Song, P. Achananuparp, E.-P. Lim, and X. Li. Topical keyphrase extraction from Twitter. In *HLT '11: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011.