User-driven geo-temporal density-based exploration of periodic and not periodic events reported in social networks

Paolo Arcaini, Gloria Bordogna, Dino Ienco, Simone Sterlacchini

Information Sciences 340–341 (2016) 122–143

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

User-driven geo-temporal density-based exploration of periodic and not periodic events reported in social networks

Paolo Arcaini[a,*], Gloria Bordogna[b], Dino Ienco[c,d], Simone Sterlacchini[e]

[a] *Charles University in Prague, Faculty of Mathematics and Physics – Czech Republic*
[b] *CNR IREA, Milano – Italy*
[c] *IRSTEA, UMR TETIS, Montpellier – France*
[d] *LIRMM, Montpellier – France*
[e] *CNR IDPA, Milano – Italy*

ARTICLE INFO

Article history:
Received 11 April 2014
Revised 25 June 2015
Accepted 1 January 2016
Available online 11 January 2016

Keywords:
Crawling social networks
User Query
Density-based Clustering
Geo-temporal proximity measure

ABSTRACT

In this paper we propose a procedure consisting of a first collection phase of social network messages, a subsequent user query selection, and finally a clustering phase, defined by extending the density-based DBSCAN algorithm, for performing a geographic and temporal exploration of a collection of items, in order to reveal and map their latent spatio-temporal structure. Specifically, both several geo-temporal distance measures and a density-based geo-temporal clustering algorithm are proposed. The approach can be applied to social messages containing an explicit geographic and temporal location. The algorithm usage is exemplified to identify geographic regions where many geotagged Twitter messages about an event of interest have been created, possibly in the same time period in the case of non-periodic events (aperiodic events), or at regular timestamps in the case of periodic events. This allows discovering the spatio-temporal periodic and aperiodic characteristics of events occurring in specific geographic areas, and thus increasing the awareness of decision makers who are in charge of territorial planning. Several case studies are used to illustrate the proposed procedure.

1. Introduction

Web 2.0 applications, such as the social platforms Facebook, Twitter, Foursquare, LinkedIn, provide nowadays citizens with a direct means to spread information in the form of textual messages, images, videos, links and so on, to their communities about the most diverse topics. These messages can eventually report information and comments about events the authors have been testimonies of (e.g., natural disasters, traffic jams, protests, comments on daily news, workshops, call for jobs), which have spatio-temporal contextual information, explicitly stating where on Earth (geotags) and when (timestamp) the messages were created. These metadata (geotags and timestamp) are embedded within the content of the message, so that the analysis of an event of interest can be performed by applying a space–time classification.

Analyzing the latent spatio-temporal structure of a huge amount of messages, reporting information or comments on an event in a social network, can aid to discover the spatio-temporal characteristic of the event, which may be either *recurring* (i.e., *periodic*) or *non-periodic*, (i.e., *aperiodic*). An example of recurring event can be the occurrence of traffic jams

[*] Corresponding author. Tel.: +420 221 914 285; fax: +420 221 914 323.
E-mail address: arcaini@d3s.mff.cuni.cz (P. Arcaini).

http://dx.doi.org/10.1016/j.ins.2016.01.014
0020-0255/© 2016 Elsevier Inc. All rights reserved.

in a specific place at specific time periods of the day; such periodic events can be expressed linguistically by a phrase such as "*every morning from 8 a.m. to 9 a.m. in Milan Central station there is traffic jam*"; such information can be used by the administration for planning alternative routes for public transports in that time interval. Another example of regularly recurring event is the presence of a huge number of tourists in a region in specific periods of the year; identifying such periods and interested areas can be useful to plan social events and provide tourist facilities. Conversely, an example of aperiodic event reported in messages can be related to the occurrence of an extraordinary natural disaster like the typhoon Haiyan in Philippines in 2013, whose analysis can benefit from the exploitation of the information contained in messages, to better understand the human perception and the consequences of the event. Another example of aperiodic long term event could be a social/political crisis, where the long term analysis of social networks can improve the awareness of the impact on the conditions of the populations and reasons of outbreaking of the crisis in specific geographic areas.

Providing means to explore the unknown spatio-temporal distribution and density of messages reporting information on events of interest can increase our ability to make predictions in several contexts about similar events that may occur in the future, and may support territorial and social planning and analysis.

In this paper, we propose a two steps procedure for analyzing events of interest to a user reported in social networks: firstly, messages about events of possible interest are collected based on crawling the social networks; in the second step, the user can specify some criteria in order to drive an original geo-temporal analysis of the messages dealing with an event of interest to verify an "a priori" hypothesis. In this second step, first a user queries the collection to filter messages about an event of interest that he/she wants to study, and finally a density-based clustering defined in the paper is applied for performing a geographic and temporal exploration of the filtered messages in order to reveal and map their latent spatio-temporal structure.

Our proposal is innovative since it is flexible for several aspects: it is user-driven, and allows making several geographic and temporal explorations to discover periodic and aperiodic events, at local or global scale.

After the seminal work on the evolution of topics in the news sphere [19], many approaches have been proposed which allow us to explore the contents of the messages created within social media such as Twitter, in order to analyze the polarity about some themes [8,22], and to identify the initiators and followers by applying in and out link analysis [33].

Other approaches have also considered the problem of geographic or temporal analysis of tweets, like in [16] where the authors' goal is to characterize the geographical spread of a communication network, in [34] where the authors examine information spread in the social network and across geographic regions, in [26] and [32] where the authors analyze the temporal spread and evolution of news stories, and in [17] where the temporal analysis of retweets is studied. Very few approaches have considered the problem of geographic and temporal analysis, like in [2] where temporal and spatial analyses are performed investigating the time-evolving properties of tweets and the geospatial characteristics of highly popular topics, or in [15], where the authors study the geographic distribution and propagation of tweets hashtags and the peak characteristics of their temporal distribution. These approaches mainly explore the geographic and the temporal correlations with contents of tweets separately, but do not apply geographic-temporal analysis.

For example, *Twitris* is an interactive web application that maps temporal summaries of tweets about a number of selected critical geographic natural and social events, such as Oklahoma tornado, India Floods, and "Occupy Wall Street" protests. The user can select the time period and one of the listed events to see the summary of the tweets about it within the time period, but does not apply spatial analysis of the tweets. Conversely, in [3] the spatio-temporal dynamics of tweets about the same topic are explored with a fixed timestamp of 1 h.

On the other side, there are crowdsourcing platforms accessible over the Internet that allow citizens to freely create messages of events they have been testimonies of, in the form of Volunteered Geographic Information (VGI) [7,12–14]: such applications perform some spatial and temporal analysis of the messages [1,5,21]. For example, the *Ushahidi* application [23] allows applying spatial clustering depending on the visualization scale of the map to group messages close in space, but completely disregards their content and timestamp. It also allows the temporal tracking of the frequency of VGI messages that deal with a given content, identified by a term within the textual message. This allows tracking the temporal evolution of a topic "popularity", irrespective of the geographic location.

In the recent approach in [10], a space–time scan statistics method to analyze tweets is proposed. It looks for clusters within the dataset across both space and time, regardless of the tweet content. By this approach, it is expected that clusters of tweets will emerge during spatio-temporal relevant events, as people will tweet more than expected in order to describe the event and spread information. The authors apply this approach to identify a disaster in London. Nevertheless, when different events occur in the same spatio-temporal region, this approach cannot tell them apart.

The objective of our proposal is defining a flexible user-centered exploratory framework for performing both geographic, temporal, and geo-temporal analysis of a collection of items having an explicit geo-temporal reference. The study of the spatio-temporal pattern of geotagged tweets has been recognized as providing important information for various applications, such as urban science, location-based services, targeted advertising, content delivery networks, and social media research [15].

The originality of our proposal is the flexibility of the approach that allows a user to specify both **what** is the event of interest he/she wants to explore, and **how** he/she intends to explore it, by choosing a temporal, geographic, or geo-temporal analysis. Specifically, the proposal permits to identify both geographic events occurring in specific regions (but not characterized by a specific timestamp such as social and economic crises during a long time period) and temporal events diffused at a global scale on Earth and characterized either by a recurring periodic timestamp or not; finally, the proposal allows to identify geo-temporal aperiodic or recurring periodic events.

As far as we know, there is not a similar flexible approach for a user-driven geo-temporal exploration of social networks.

A possible use of our approach is to identify the geographic areas where there are many messages created by citizens about an event, for example the soccer world cup championship, in order to analyze where soccer is more popular.

Conversely, another possible use is identifying if an event, as a tennis match, is reported in many messages created simultaneously, for example during the tennis match itself: this could give an indication of how many people were looking at the tennis match on TV at a global scale, i.e., irrespective of their geographic location.

Another use of the approach is to identify messages that have been created simultaneously in a given region about an event, for example to identify the geographic regions where a sport match on TV has obtained the highest audience.

Finally, our proposal permits to identify events that are periodically reported in messages created in geographic regions at specific time periods, as *every day*, *every week*, *every month*, etc. This may be useful to detect messages reporting recurring events such as traffic jams in the morning and evening in specific streets, which can reveal and help understand human behaviors (for example, the attitude to complain about traffic jam more in the evening rather than in the morning) and the geographic-dependent life style patterns of citizens (for example, different rush hours depending on the region).

The exploratory functions we propose for carrying out geo-temporal analysis of recurring and aperiodic events reported in geotagged messages are defined based on different instantiations of an original density-based geo-temporal clustering algorithm, named *GT-DBSCAN*. *GT-DBSCAN* is defined in this paper as an extension of the classic DBSCAN algorithm [11] and it uses distinct geo-temporal distance measures defined in the paper too. Notice that the clustering algorithm can group entities based on a geographic distance measure, instead of a Euclidean distance, so as to better model events occurring at a global scale on Earth.

Although this density-based clustering has been conceived for a geo-temporal analysis of social network messages, it must be outlined that it can be applied to analyze any collection of items having geotags and timestamp.

In the paper, in Section 2 we review the related literature. Section 3 presents the framework we are proposing, and Section 4 introduces some functions we use to derive the temporal features from the social messages. Section 5 describes the proposed density-based geo-temporal clustering algorithm and some geo-temporal measures. Section 6 discusses some results obtained by the application of such functions, and the conclusions in Section 7 summarize the main achievements.

2. Related literature

Our proposal is related to both the problem of spatio-temporal mining of events in social networks, and density-based clustering algorithms for spatio-temporal data.

Until now spatio-temporal mining in social networks has been intended with the purpose of either identifying groups of people whose activities on the social network co-occur in space and time or to identify frequent co-occurring tags in social media like Flickr [35].

Our meaning is different and refers to *detecting events*, reported by many people either close in space or in time, or even both in space and time, within social networks as in [28] where detection of earthquakes is done considering Twitter as a social sensor.

Events in social networks can be identified by analyzing the spatial-temporal and content information of the messages. Events might have active participants, and they are of large scale, that is many users experience them; moreover, they may influence people's life and this is why they are reported in many messages in the same or close periods of time in social networks. Thus, events have a temporal reference [25], and generally also have a spatial reference, so that also their location can be identified. Such events include social events such as economic crisis, sports events, exhibitions, accidents, political campaigns, and also natural events such as storms, hurricanes, and earthquakes.

In [28], an approach to use Twitter messages (*tweets*) to identify events is proposed based on the assumption that each Twitter user is regarded as a sensor and each tweet as sensory information affected by noise. Thus, event detection is reduced to the problem of object detection having spatial and temporal references and is approached based on the application of an SVM classifier.

A simplification of the problem of mining events in social media is the assumption that nowadays, in most of the social networks, the geo-location of the creators of the message is captured by the GPS device from which the message is sent and is encoded by a geotag in the message, and the time of the creation of the message from a mobile device is the time of observation of the event dealt with in the message.

Although messages with explicit geotags are still a minority of the whole messages exchanged within social contexts, the growing diffusion of smart devices equipped with GPS sensors allows us to guess that they will grow in the near future.

For example, since 2009, Twitter released the location service that enables mobile users to publish their tweets with latitude and longitude. Our approach needs that messages have explicit geographic and temporal references, as in [18]. Other approaches, like [9], extract from the message the geographic reference; however, this is not the topic of the current paper, and so we only consider geotagged messages. In fact, the focus of this paper is to define a flexible procedure to collect and cluster messages about an event of interest based on their geo-temporal attributes. Thus, the applicability of the approach goes beyond the scope of social network analysis.

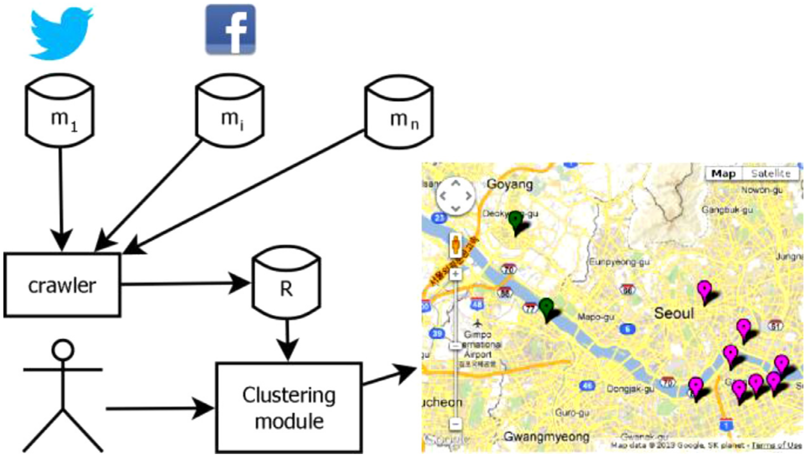


Fig. 1. Proposed approach for a flexible geo-temporal exploration of social media messages.

A further distinction on event detection in social contexts can be made based on the fact that the detection must be executed real time, as early as possible with respect to the event occurrence, or not. The first approach needs to filter a stream of messages [28,30,31], while the second one can rely on querying a repository of the messages, as in our case.

In order to identify groups of messages which have geotags close in space, some approaches adopt DBSCAN algorithm, such as in [18], for its ability to detect clusters of any shape, which is one of the characteristics of the close geotags of messages whose geographic reference can even change over time. This is also one of the motivations of our proposal of extending the DBSCAN clustering algorithm in order to evaluate a geo-temporal distance measure between the geotagged and time-stamped messages. Many density-based clustering algorithms have been defined, such as [4,24,20], suitable to detect groups of objects whose spatial or spatio-temporal distribution can have different shapes, but they have been applied in other data mining context applications different from social networks.

As far as we know, no approaches have been proposed for an integrated geographic-temporal analysis of messages in social contexts, while most approaches mainly explore separately the geographic and the temporal relations among the messages. Moreover, no clustering algorithm has been defined for identifying recurring periodic geographic events that are of interest for some application context, such as the identification of the streets and the hours of the day where “traffic jams” occur regularly, which can help territorial planning.

Our approach takes inspiration from the Spatio-Temporal DBSCAN (ST-DBSCAN) clustering algorithm [4] in order to define a geo-temporal DBSCAN (GT-DBSCAN). While the DBSCAN algorithm uses only one distance to measure the closeness of data in space (generally the Euclidean distance), ST-DBSCAN supports two dimensional spatial data, by defining the similarity based on the conjunction of two density tests, one based on the spatial distance and the other one based on the temporal distance.

As discussed in the following section, we defined integrated spatio-temporal distance measures as in [24]. We replace the Euclidean distance with a *geographic* distance to detect an event at a global scale, extend the temporal distance so as to define also a *modulo temporal* distance, in the case one wants to detect recurrent periodic events, and combine the geographic and temporal dimensions in a non-compensative way so as to achieve a clear interpretation of the generated clusters.

Thus, our GT-DBSCAN proposal is innovative not only for the geo-temporal distance it uses, but also because it can identify geo-temporal clusters of periodically recurrent events. We defined the algorithm by extending the DBSCAN algorithm (and not ST-DBSCAN) because several distance measures can be employed which can neglect either the geographic or the temporal dimension, thus performing only a temporal analysis at a global scale or geographic analysis of long term events.

3. Exploratory framework of social media

The proposed procedure is depicted in Fig. 1.

We assume that a set of social sources of information exist, providing web services that allow a user to retrieve from the social sources, through a query, the messages dealing with a given content of interest.

Let $S = \{s_1, \ldots, s_n\}$ be a set of (social) information sources (social platforms). Each $s_i \in S$ is represented as follows:

$$s_i = \{n_i, M_i, ws_i\} \tag{1}$$

where n_i is the name of the information source, M_i the set of stored messages, and ws_i a web service provided by the information source for querying and retrieving subsets of M_i. Notice that each s_i (specifically, each ws_i) has its own query language Q_i so that each user query q must be translated into a query $q_i \in Q_i$ when querying source s_i.

For the sake of simplicity, we assume that each message $m \in M_i$ is represented as a tuple of fields as follows:

$$m = \langle f_1, \ldots, f_{\mathrm{f}} \rangle \tag{2}$$

where each f_i takes values on a domain D_i.

We require that $\forall s_i$ at least three mandatory fields exist in each *m,*

- a geotag field $f_s = (lat,lon)$, where $(lat,lon) \in R \times R$ are the geographic coordinates of the location from which the message has been sent;
- a temporal field $f_t \in Date$ defining the timestamp (i.e., the date)of the message submission; f_t is represented in ISO 8601 format

  $$\mathrm{YYYY - MM - DD'T'hh : mm : ss'Z'[\pm hh : mm]} \tag{3}$$

  where *Z* refers to the UTC time zone, and ±*hh:mm* is the offset in hours and minutes with respect to UTC relative to the time zone containing the coordinates (*lat,lon*);
- finally, a textual field $f_c \in \{string,[string]\}$containing one or more keywords identifying the semantics of the message content; they can be the *hashtags* in tweets, and terms extracted from the textual part of the messages.

Thus, each message *m* is represented by a *report* $r \in R$ with $r = <f_s, f_t, f_c, url>$, where *url* is a unique resource location that uniquely identifies the report on its social platform.

The first phase of the procedure consists in extracting from the *n* sources $s_i = \{n_i, M_i, ws_i\}$ (with $i = 1,\ldots, n$) the messages that deal with potential events.

To this end, a crawler is run so as to periodically submit a set *Q* of requests to the *n* sources, starting from a specified initial date, with a given period (for example, at each hour or each day) until an ending date is reached. This consists in first translating each query in *Q* into the *n* similar queries q_i according to the syntax of the sources (similar with respect to their meaning) and then in periodically invoking the web services ws_i of the targeted sources with the translated queries q_i,so as to retrieve only the messages whose content satisfies q_i. The retrieved messages are parsed to extract their geotag field f_s, temporal field f_t, and content field f_c, and are added to the collection $R(Q)$ of the messages satisfying the set of queries. When the ending date is reached, the collection $R(Q)$ is ready for the subsequent user-centered exploration.

A user can select a subset of reports dealing with a topic of his/her interest by querying the collection $R(Q)$ through the specification of a query *q* defined as the conjunction of a content based, a spatial based, and a temporal based condition. The content based condition is expressed by a list of terms that are evaluated against f_c as in the vector space model; the spatial based condition is expressed by a pair of geographic coordinates defining the low left and up right corners of the interesting geographic Bounding Box (*BB*) where the field f_s of the reports must be contained, thus the implicit spatial operator is $f_s \subset BB$; the temporal based condition specifies the time range $[t_{start}, t_{end}]$ delimiting the lower and upper creation dates of the reports of interest; the implicit temporal operator is $f_t \in [t_{start}, t_{end}]$.

The reports in $R(q)$ that satisfy the user query are finally clustered by applying the proposed GT-DBSCAN clustering algorithm with an indication of the desired distance measure to use, which depends on the kind of analysis one intends to perform, as described in Section 5.

4. Time representation

In order to deal with time in the clustering process, we adopt a homogenous representation of the temporal component [6] that permits to arrange *events* on a directed numeric time line with a given granularity, so that the computation of the time distance between two events is possible.

We assume as origin of the time line the Unix *epoch*, hereafter indicated by d_O = 1970-01-01T00:00Z, and the finest granularity the *seconds*, so that the time distance between two events is expressed in seconds.

Therefore, when the time granularity in *seconds* is appropriate for the analyses we are carrying out, we have to convert each temporal field f_t of each report (given in ISO 8601 format) into a point of the time line, that is obtained by computing the distance in seconds of the date f_t from the time origin d_O.

Nevertheless, for some events and purposes it may be irrelevant to evaluate their time distance from the origin in seconds. For example, in the case of messages about a sport match, it may be interesting to know if they are created when the match has been shown on TV: in this case, the hour could be chosen as appropriate granularity, so that the messages sent during the match have the same time distance from the time origin of the match. To this end, we introduce the notion of time unit.

4.1. Time unit

Since one can be interested in considering events using distinct time granularities, we introduce the notion of time unit *G*.

A time unit *G* is always multiple of the basic time unit (G_s, the time unit of seconds) and possibly also of another not basic time unit. For example, the time unit of minutes is $G_m = 60G_s$, and the time unit of hours is $G_h = 60G_m = 3600G_s$.

A time unit G determines a partitioning of the time line into consecutive *granules*; each granule contains events which are not discernible assuming G as time unit. For example, events related to timestamps 2013-07-24T14:36:05Z and 2013-07-24T14:36:48Z are different when using the basic time unit of seconds G_s, whereas they are indiscernible when the time unit is minutes G_m.

4.2. Time point

Given a time unit G, we identify a *time point* with $[t, G]$, where t is the number of granules of unit G from the time origin (the Unix epoch). In order to compute a time point starting from a date (as defined in (1)) and a time unit G, we introduce two functions.

4.2.1. Function getTimePointUTC

Given a temporal field $c \in Date$ as defined in (3) and a time unit G, function *getTimePointUTC* computes the number of granules of type G contained between f_t and the time origin d_O, i.e.,

$$getTimePointUTC\ :\ Date \times \{G_s, G_m, G_h, \ldots\} \rightarrow N \tag{4}$$

Notice that this function takes into account all the information in f_t (as defined in (1)) by transforming it in the equivalent date relative to the UTC timezone: so, given two *simultaneous* dates in different time zones, function *getTimePointUTC* returns the same result expressed in a desired time unit G.

For example, given the following dates:

2013-07-23T05:45:20Z (time zone *UTC*)
2013-07-23T14:45:20Z + 09:00 (time zone *Korea Standard Time*)
2013-07-22T22:45:20Z – 07:00 (time zone *Pacific Daylight Time*)
2013-07-22T22:45:25Z – 07:00 (time zone *Pacific Daylight Time*)

by applying *getTimePointUTC* with time unit G_s (i.e., seconds), we obtain the same result 1374558320 for the first three dates, and the value 1374558325 for the last date. Instead, with the time unit G_y (i.e., year) we obtain the same result 43 for all the four dates. This exemplifies how we can perform temporal analyses by exploiting the information granule G in order to be more or less precise.

This transformation is used when one is interested in grouping simultaneous events, such as messages related to a live sport event at a global scale.

4.2.2. Function getTimePointTZ

Function *getTimePointTZ* is defined as follows:

$$getTimePointTZ\ :\ Date \times \{G_s, G_m, G_h, \ldots\} \rightarrow N \tag{5}$$

Notice that this function, given a date f_t represented as in (3), disregards the offset information in f_t ([±*hh:mm*] after *'Z'*) and assumes the time zone UTC. Furthermore, it truncates the resulting date based on the specified information granule. For example:

- dates 2013-07-23T14:45:20Z + 09:00 (time zone *Korea Standard Time*) and 2013-07-23T14:45:20Z (time zone *UTC*)are considered equivalent to 2013-07-23T14:45:20Z;
- dates 2013-07-22T22:45:20Z – 07:00 (time zone *Pacific Daylight Time*) and2013-07-22T22:45:20Z-10:00 (time zone *Cook Island Time*) are considered equivalent to 2013-07-22T22:45:20Z.

In this case, the application of *getTimePointTZ* to the four dates in the example in Section 4.2.1 yields different results:

getTimePointTZ(2013-07-23T05:45:20Z,G_s) = 1374558320
getTimePointTZ(2013-07-23T14:45:20Z + 09:00,G_s) = 1374590720
getTimePointTZ(2013-07-22T22:45:20Z – 07:00,G_s) = 1374533120
getTimePointTZ(2013-07-22T22:45:25Z – 07:00,G_s) = 1374533125

We can see that, for the dates not belonging to UTC, the computed value is different from the value computed by function *getTimePointUTC*. More precisely, the difference between the values computed for the same date by the two functions is given by the offset in hours and minutes with respect to the UTC.

Function *getTimePointTZ* is used when one wants to group events that are not necessarily simultaneous at a global level, but that occur in close units of time (depending on the chosen granularity) relatively to their time zone. For example, if one wants to group events happening in close hours and minutes, he/she may use *getTimePointTZ* by specifying as time unit G_m, so that events occurring in the same hour of the same day and same minutes relatively to their time zone will get the same point on the time line, while events occurring in different hours of the day (e.g., one in the morning and one in the afternoon relatively to their time zone) or same hour but different minutes will appear in two distinct clusters of points on the time line.

Moreover, if one wants to identify periodically recurrent events occurring, for example, at the same hour of the day, irrespective of the day, month, and year, he/she has to use the functions defined in the following.

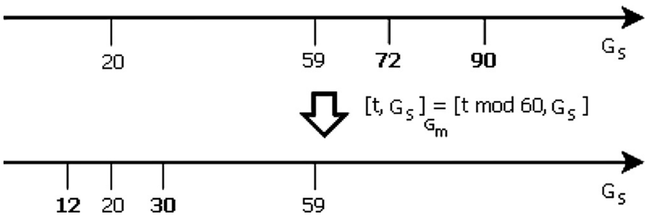


Fig. 2. Example of application of the modulo operation.

4.3. Time interval

With $[\Delta n,G]$ we identify a time interval of n granules of the time unit G. Note that the time interval is not related to the time origin. Moreover, notice that a time interval (as a time point) is a multiple of the basic time granule of seconds G_s (i.e., $[\Delta\ n,G] = [\Delta\ mn,G_s]$, with $m \in N$ and $G = m{\cdot}G_s$).

4.4. Period

Given a time line with time unit G, we can specify another time unit named *period,* indicated by $\tilde{G}$ with $\tilde{G}=$ n · G ($n \in N$ and $n > 1$), so that any time point $[t,\ G]$ of the time line can be mapped into $[t,\ G]\ \tilde{G}$ by applying the modulo operation as follows:

$$[t,G]_{\tilde{G}} = [t\bmod n, G] = [r, G] \ \text{ with } \tilde{G} = n \cdot G \ \text{ and } r = t - n^{*}\text{trunc}(t/n) \tag{6}$$

where trunc(.) removes the fractional part of its argument.

Fig. 2 shows an example of application of modulo G_m to a time line with time unit G_s.

In the first line we have the time points on the time line using the granularity G_s, i.e., seconds. The vertical arrow represents the modulo operation, defined in formula (6), that transforms the time points expressed in seconds into time points where the period of 60 s, i.e., 1 min, has been subtracted. The time points 72 and 90 are respectively mapped into 12 and 30 in the bottom time line.

This transformation can be useful to identify events periodically recurring with a period $\tilde{G}$.

For example, if we were interested in social messages related to the *traffic jam* occurring daily, we should only consider the *hours*, *minutes*, and *seconds* of the message, but not the *year, month,* and *day*; in this case, the period of analysis would be the *day*.

The modulo operation can be applied also to time intervals in the following way:

$$[\Delta t, G]_{\tilde{G}} = [\Delta(t\bmod n), G] = [\Delta r, G] \quad \text{with} \quad \tilde{G} = n \cdot G \quad \text{and} \quad r = t - n * \text{trunc}(t/n) \tag{7}$$

5. Geo-temporal density-based clustering GT-DBSCAN

In this section, we illustrate the last phase of the proposed approach, which is the density-based clustering of the messages about an event of interest, retrieved by the user query in the preceding phase.

The objective of the geo-temporal density-based clustering is to generate groups of messages by evaluating the geographic-temporal distance of their representations as introduced in Section 3.

Specifically, this phase can be tuned to support several distinct kinds of analyses:

- grouping messages sent by *close* geographic locations on Earth, neglecting their timestamps, so as to explore the events from long term, possibly covering a time range wider than the range in which the crawler of the first phase was run to collect the messages. An example of such analysis is to explore the geographic regions more interested by an economic, political or social crisis;
- grouping messages *simultaneously* sent, neglecting their geotags, to analyze events at a global scale, such as the popularity of soccer matches of the world cup championship;
- grouping messages sent at recurring timestamps with a given period, neglecting their geotags, to analyze periodic events at a global scale, such as to identify the hours of the day in which traffic jams are reported more often on Earth;
- grouping messages *simultaneously* sent by *close* geographic locations on Earth to analyze global events at a local scale, such as the popularity of the soccer matches of the world cup championship in each region;
- grouping messages sent by *close* geographic locations on Earth at recurring timestamps with a given period $\tilde{G}$ to analyze, for example, the hours when most often traffic jams occur in each region.

The assumptions of our approach are the following:

- one must have a hypothesis to test, i.e., an idea, although vague, of the duration and of the periodicity of an event he/she intend to analyze, and an idea of the spread of the event at a local or a global scale. For example, in the case of sport events, the meaningful granularity for the analysis could be hours and not the minutes; in the case of pandemic events the granularity of the analysis could be the day since considering hours could be meaningless. The soccer world cup championship can be assumed popular at a global scale, while “a pop singer tour” can be object of many local comments;

- one does not know the shape of the groups;
- each group might have a different shape;
- the number of groups is not known in advance; they may be many or a few;
- we want to apply some density-based clustering, since the assumption is that many messages on the same topic, generated in a close area and/or sent simultaneously or at periodic timestamp, are indications of the popularity of some event.

To this end, we need to apply some kind of unsupervised spatio-temporal mining approach using a flexible density-based criterion to drive the grouping, allowing in this way to test the hypothesis.

Density-based clustering algorithms have a wide applicability in unsupervised spatial data mining. They apply a local criterion to group objects: clusters are regarded as regions in the data space where the objects are dense, and which are separated by regions of low object density (noise). Among the density-based clustering algorithms, DBSCAN [11] is very popular due to both its low complexity and its ability to detect clusters of any shape, which is a desired characteristic when one does not have any knowledge of the possible clusters' shapes, or when the objects are distributed heterogeneously with different shapes, such as along paths of a graph or a road network (as in one of our experiments). Furthermore, DBSCAN does not need as input the number of clusters to generate, which is indeed unknown in our case. Starting from the DBSCAN, some spatio-temporal density-based clustering algorithms have been defined, such as ST-DBSCAN in [4], where first the temporal neighbors are filtered and then the DBSCAN algorithm is applied to form the clusters. In this approach, space and time dimensions are not analyzed in an integrated manner, as it also happens with the STSNN spatio-temporal clustering algorithm [20]. Conversely 4D+SNN [24] is a spatio-temporal clustering algorithm that considers all dimensions simultaneously in the distance measure. The distances on each dimensions are aggregated in a compensative way (by a weighted sum) so that the spatio-temporal properties of the generated clusters may have different characteristics, i.e., very close in space or time, or even averagely close both in space and time. So, by this aggregation the interpretation of the whole results is cumbersome and does not shade light on the nature of the detected events. A cluster might be generated either because its reports have been created by close geographic regions in a long period of time or because its reports are created in close dates from distant regions. Thus one cannot verify the hypothesis on the nature of the event because the weighted sum is not enough specific for the purposes of the geo-temporal exploration.

Taking ideas both from ST-DBSCAN and 4D+SNN, we defined the Geo-Temporal DBSCAN (GT-DBSCAN) clustering algorithm. In order to introduce its definition, we recap the classic DBSCAN algorithm.

5.1. Classic DBSCAN algorithm

Unlike other clustering algorithms, DBSCAN does not require the number of clusters as input, but it needs a pair of parameters defining the density of the clusters in terms of *minimum number of elements* (*minPts*) that must exist within a *given distance* (ε) from an element in order to start forming a cluster.

More specifically, the DBSCAN algorithm assigns elements represented by points on a spatial domain to particular clusters or designates them as statistical noise if they are not enough close to other points. DBSCAN determines cluster assignments by assessing the local density at each point using two parameters: a reachability parameter which is a distance value (ε), and the minimum number of points (*minPts*).

A point p which meets the minimum density criterion (namely there are at least *minPts* other points p_i located within distance ε from p) is designated as *core* point. Formally, given a set $P = \{p_1, p_2, \dots p_M\}$ of M points p_{i} defined on the 2-dimensional domain R^2 (i.e., $p_i = (x_i{,}y_i)$), $p \in P$ is a core point if at least a minimum number *minPts* of points $p_j \in P$ exists such that $||p_j - p|| < \varepsilon$, where ||.|| is the Euclidean distance.

Two core points p_i and $p_{j,}$ with $i \neq j$ such that $||p_i - p_j|| < \varepsilon$, define a cluster c ($p_i, p_j \in c$) and are defined as *core points* of c, i.e., $p_i, p_j \in \mathrm{core}(c)$. All non-core points within the maximum distance ε from a core point are considered as *non-core members* of a cluster, and are called *boundary* (or *border*) points: $p \notin \mathrm{core}(c)$ is a boundary point of c if $\exists \mathrm{p_i} \in \mathrm{core}(c)$ with $||p - p_i|| < \varepsilon$.

Finally, points that are not part of any cluster are considered as *noise*: $p \in P$ is a noise point if $\forall c, p \notin \mathrm{core}(c)$ and $\neg\exists p_i \in \mathrm{core}(c)$ with $||p - p_i|| < \varepsilon$.

5.2. GT-DBSCAN algorithm

The Geo-Temporal DBSCAN algorithm (GT-DBSCAN) is defined by extending the classic DBSCAN described in the previous subsection, to allow the evaluation of distinct distance measures.

The algorithm is defined as follows. Given a set $R(q)$ of n reports $r_{i,}$ possibly about the topics of a query q, we represent each of them by a point p_i in a three dimensional geographic and temporal space so that $p_i = (x_{i1}; x_{i2}; x_{i3})$, with x_{i1} and x_{i2} corresponding to the geographic coordinates latitude and longitude, and being $x_{i3} = f_t$ the timestamp.

A point $p \in R \times R \times R^{+}$, associated with a report $r \in R(q)$, is a core point if at least a minimum number *minPts* of points p_{j} (associated with reports in $R(q)$ as well) exists such that

$$\mathrm{Satisfy}(\mathrm{Distance}, p, p_j, \varepsilon_1, \varepsilon_2, G, \tilde{G}) = \mathrm{True} \tag{8}$$

where:

- Distance ∈ {DistG, DistT, DistMT, DistGT, DistGMT} is the name of a distance measure that the algorithm can use, namely: Geographic distance (DistG), Temporal distance (DistT), Modulo-Temporal distance (DistMT), Geo-Temporal distance (DistGT), Geo-Modulo-Temporal distance (DistGMT). These distance measures are defined in the following subsections;
- G and $\tilde{G}$ are time units with $\tilde{G}$= nG being a period of time;
- ε_1 and ε_2 are numeric values whose domains depend on the parameter Distance.

Satisfy(.) is a binary function that returns *True* when the points are within a geographic distance ε_1 and a (modulo)-temporal distance ε_2, while it returns *False* when at least one of the previous conditions is not satisfied.

Two core points p_i and p_j, with $i \neq j$ and such that *Satisfy*(*Distance*, p_i, p_j, ε_1, ε_2, G, $\tilde{G}$) = *True*, define a cluster c (i.e., p_i, $p_j \in c$) and are core points of c, i.e., p_i, $p_j \in \mathrm{core}(c)$.

A non-core point p is a boundary point of a cluster c if $p \notin \mathrm{core}(c)$ and $\exists p_i \in \mathrm{core}(c)$ with *Satisfy*(*Distance*, p, p_i, ε_1, ε_2, G, $\tilde{G}$) = *True*.

Finally, points that are not part of a cluster are considered as noise: p is a noise point if $\forall c$, $p \notin \mathrm{core}(c)$ and $\neg\exists p_i \in \mathrm{core}(c)$ with *Satisfy*(*Distance*, p, p_i, ε_1, ε_2, G, $\tilde{G}$) = *True*.

A value *Distance* ∈ {*DistG, DistT, DistMT, DistGT, DistGMT*} can be used in an instantiation of the *GT-DBSCAN* algorithm as follows:

$$\mathrm{GT-DBSCAN}(\mathrm{Distance}, \varepsilon_1, \varepsilon_2, \mathrm{G}, \tilde{\mathrm{G}}, \mathrm{MinPts}) \tag{9}$$

with *MinPts* ∈ N. Note that the type of distance influences the kind of ε_1 and ε_2 that must be used. In the following, we describe the five distance measures.

5.3. Geographic distance (DistG)

The geographic distance is used for discovering reports that have been submitted by places that are in close geographic regions.

In order to compute the geographic distance between two reports, we use an Earth-based distance, i.e., the *haversine formula* [29] which is, for our purposes, a good approximation of the real geographic distance on Earth. The *DistG* measure is defined based on the *haversine* formula as follows:

$$DistG(r_a, r_b, \varepsilon) = \frac{2\rho \arcsin\left(\sqrt{\sin^2\left(\frac{lat_b - lat_a}{2}\right) + \cos(lat_a)\cos(lat_b)\sin^2\left(\frac{lon_b - lon_a}{2}\right)}\right)}{\varepsilon} \tag{10}$$

where the numerator corresponds to the harversine formula, ρ is the Earth's radius, r_a and r_b are the pairs of geographic coordinates of points a and b, $r_a = (lat_a, lon_a)$ and $r_b = (lat_b, lon_b)$ and $\varepsilon \in R^{+}$.

Notice that *DistG* takes values in R^{+}: $DistG(r_a, r_b, \varepsilon) \in [0,1)$ when the harversine formula is smaller than ε, otherwise $DistG(r_a, r_b, \varepsilon) \geq 1$. Moreover, $DistG(r, r, \varepsilon) = 0$, it is symmetric (i.e., $DistG(r_a, r_b, \varepsilon) = Dist(r_b, r_a, \varepsilon)$), and respects the triangle inequality(i.e., $DistG(r_a, r_b, \varepsilon) + DistG(r_b, r_c, \varepsilon) > DistG(r_a, r_c, \varepsilon)$ for $r_a \neq r_b \neq r_c$). This formula gives only an approximation of the real distance between two points on Earth, due to the its non-perfect spherical shape, and, when r_a and r_b are far one another, approaching half the Earth circumference, a small error is often not a major concern. For analyses at a local scale, when the Earth curvature is not influential, we can use the Euclidean distance; in this case, GT-DBSCAN reduces to the classic DBSCAN.

When ε tends to 0, *DistG* tends to infinity, while, on the contrary, when ε is bigger than the Earth circumference, *DistG* is smaller than 1.

If the reports are originated from a network (e.g., a street network) one could use a network-based distance; for the distances on a street network, the Distance Matrix Service of the Google Maps JavaScript API[1] could be used.

5.3.1. Geographic epsilon

When using the geographic distance, the *GT-DBSCAN* algorithm is invoked as *DBSCAN*(*DistG*, ε_s,∝, *G*, $\tilde{G}$, *MinPts*), where $\varepsilon_s \in R^{+}$ identifies a distance in km between two reports, ∝ is an arbitrary large positive number, and G and $\tilde{G}$ are two arbitrary time granules.

In this case, $Satisfy(DistG, r_a, r_b, \varepsilon_s, \propto, G, \tilde{G}) = (DistG(r_a, r_b, \varepsilon_s) < 1)$, that takes the value *True* when the geographic distance between the two reports is smaller than ε_s.

[1] https://developers.google.com/maps/documentation/javascript/distancematrix

5.4. Temporal distance (DistT)

The temporal distance is used for discovering reports that have been submitted in simultaneous or close time points.

The user can specify a time unit G, otherwise the standard unit of seconds G_s is used. Given a time unit G, the time points of two reports r_a and r_b (whose dates are f_{ta} and f_{tb}) are t_a and t_b, computed using function *getTimePointUTC* as defined in formula (4). The temporal distance between r_a and r_b is defined as the one norm distance in the one dimensional space:

$$DistT(r_a, r_b, G, \varepsilon) = \frac{[\Delta|t_a - t_b|, G]}{\varepsilon} = \frac{|[t_a, G] - [t_b, G]|}{\varepsilon} \qquad \text{with } \varepsilon \in N \tag{11}$$

Notice that $DistT(r,r,G,\varepsilon) = 0$, $DistT(r_a,r_b,G,\varepsilon) = DistT(r_b,r_a,G,\varepsilon)$, and $DistT(r_a,r_b,G,\varepsilon) + DistT(r_b,r_c,G,\varepsilon) \geq DistT(r_a,r_c,G,\varepsilon)$. Furthermore, as ε tends to infinity, $DistT(r_a,r_b,G,\propto)$ tends to zero, while, when ε tends to zero, *DistT* tends to infinity. Moreover, $DistT(r_a,r_b,G,\varepsilon) < 1$ when the absolute difference between the time points t_a and t_b is smaller than ε, otherwise $DistT \geq 1$. Notice that, applying *DistT*, GT-DBSCAN reduces to the classic DBSCAN using a one-dimensional block distance.

5.4.1. Temporal epsilon

When using the Temporal distance, the *GT-DBSCAN* algorithm is invoked as $DBSCAN(DistT, \propto, \varepsilon_t, G, \tilde{G}, MinPts)$, where $\varepsilon_t \in N$ identifies the number of granules of the adopted time unit G between two time points associated with two reports, $\propto$ is an arbitrary large positive value, and $\tilde{G}$ is any time period. In this case, $Satisfy(DistT, r_a, r_b, \propto, \varepsilon_t, G, \tilde{G}) = (DistT(r_a, r_b, G, \varepsilon_t) < 1)$, that is *True* when the timestamps of the two reports differ for a number of time granules G less than ε_t.

5.5. Modulo-Temporal distance (DistMT)

The modulo-temporal distance is used for discovering reports that have been submitted in the same or close time points relative to a given period; the time points are rearranged with respect to a given period $\tilde{G}$.

A period $\tilde{G} = n \cdot G$ (with $n \in N$ and $n > 1$) must be specified. One can specify also the time unit G, otherwise the standard unit G_s is used.

Given a time unit G, the time points t_a and t_b of two reports r_a and r_b are computed using function *getTimePointTZ* defined in formula (5). The Modulo-Temporal distance between the two reports r_a and r_b is defined by applying definition (7) as follows:

$$\begin{gathered} DistMT(r_a, r_b, G, \tilde{G}, \varepsilon) = \frac{Min\big([\Delta|t_a - t_b|, G]_{\tilde{G}}, [\Delta n, G] - [\Delta|t_a - t_b|, G]_{\tilde{G}}\big)}{\varepsilon} \\ \text{where } [\Delta n, G] - [\Delta|t_a - t_b|, G]_{\tilde{G}} = [\Delta(n - |t_a - t_b|), G]_{\tilde{G}} \end{gathered} \tag{12}$$

with ε€∈€N being the number of time granules of type G.

It can be noticed that also this distance measure satisfies $DistMT(r, r, G, \tilde{G}, \varepsilon) = 0$, $DistMT(r_a, r_b, G, \tilde{G}, \varepsilon) = DistMT(r_b, r_a, G, \tilde{G}, \varepsilon)$ and also the triangular inequality; moreover, it assumes a value smaller than 1 when the modulo $\tilde{G}$ difference between time points is smaller than ε time granules of type G.

5.5.1. Modulo-Temporal epsilon

When using the Modulo-Temporal distance, the *GT-DBSCAN* algorithm is invoked as $DBSCAN(DistMT, \propto, \varepsilon_{mt}, G, \tilde{G}, MinPts)$, where ε_{mt}, $\tilde{G}$ and G are used in the distance formula (12). It yields that $\tilde{G} = n \cdot G$, where n is the number of granules of time unit G contained in the period $\tilde{G}$ (see definition (6)). $\varepsilon_{mt} \in N$ (such that $\varepsilon_{mt} \leq n$) identifies the distance between two points, in terms of number of granules of the adopted time unit G. $\propto$ is an arbitrary large positive value. In this case, $Satisfy(DistMT, r_a, r_b, \propto, \varepsilon_{mt}, G, \tilde{G}) = (DistMT(r_a, r_b, G, \tilde{G}, \varepsilon_{mt}) < 1)$.

5.6. Geographic-Temporal distance (DistGT)

The Geographic-Temporal distance is used for discovering reports that are close both in space and time.

The user must specify a spatial epsilon ε_s and a temporal epsilon ε_t, as described in Sections 5.3.1 and 5.4.1. The user can also specify a time unit G, otherwise the standard unit G_s is used.

Given two reports r_a and r_b, a Geographic-Temporal dissimilarity measure (that does not satisfy the triangular inequality) is defined as follows:

$$DissGT(r_a, r_b, G, \varepsilon_s, \varepsilon_t) = Max(DistG(r_a, r_b, \varepsilon_s), DistT(r_a, r_b, G, \varepsilon_t)) \tag{13}$$

By using *Max* as aggregation of the two distance measures, we want a dissimilarity that considers both the Geographic distance and the Temporal distance.

As proposed in [27], we introduce the rectifier function $f(t, \beta) = t^{\beta}$, with $t \in R_{\geq 0}$ and $b \in (0, 1]$, for modifying the non-metric dissimilarities computed by *DissGT* into metrics. A rectifier is a function $f : R_{\geq 0} \times U \to R_{\geq 0}$ that satisfies the following conditions:

(i) $U \subseteq R_{\geq 0}$ and $\inf_{u \in U} u = 0$;

(ii) $\lim_{\beta \to 0^+} f(t, \beta) = y_0 \forall t > 0,$ where $y_0 > 0$;
(iii) $f(0, \beta) = 0 \forall \beta \in U$;
(iv) f is strictly increasing in its first argument;
(v) f is sub-additive in its first argument, that is: $f(t_1 + t_2, \beta) \leq f(t_1, \beta) + f(t_2, \beta)$ for $t_1, t_2 \in R_{\geq 0}$ and $\beta \in U$.

The fourth condition of the previous ones is needed to preserve the relative order of the dissimilarities.

The Geographic-Temporal distance measure is defined as follows:

$$DistGT(r_a, r_b, G, \varepsilon_s, \varepsilon_t) = [DissGT(r_a, r_b, G, \varepsilon_s, \varepsilon_t)]^\beta \quad \text{with} \beta \in (0, 1] \tag{14}$$

It can be proven that $DistGT(r, r, G, \varepsilon_s, \varepsilon_t) = 0$, $DistGT(r_a, r_b, G, \varepsilon_s, \varepsilon_t) = DistGT(r_b, r_a, G, \varepsilon_s, \varepsilon_t)$. Furthermore, the triangular inequality is satisfied, i.e., $DistGT(r_a, r_b, G, \varepsilon_s, \varepsilon_t) + DistGT\ (r_b, r_c, G, \varepsilon_s, \varepsilon_t) \geq DistGT(r_a, r_c, G, \varepsilon_s, \varepsilon_t)$.

5.6.1. Geographic-Temporal epsilon value

When using the Geographic-Temporal distance, the *GT-DBSCAN* algorithm is invoked as *GT-DBSCAN*(*DistGT*, ε_s, ε_t, *G*, *MinPts*), where the values for the parameters ε_s and ε_t are those defined in Sections 5.3.1 and 5.4.1, respectively.

In this case, $Satisfy(DistGT, r_a, r_b, \varepsilon_s, \varepsilon_t, G, \tilde{G}) = (DistGT(r_a, r_b, G, \varepsilon_s, \varepsilon_t) < 1)$.

5.7. Geographic-Modulo-Temporal distance (DistGMT)

The Geographic-Modulo-Temporal distance is used for discovering reports that are close in space and in time within a *period*; in this distance measure, the time points are rearranged with respect to a given period.

In addition to what required by the Geographic-Temporal distance, one must also specify a period $\tilde{G}$. Moreover, instead of the temporal epsilon, one has to specify a modulo-temporal epsilon ε_{mt} as described in Section 5.5.1.

The Geographic-Modulo-Temporal distance is defined as follows:

$$DistGMT(r_a, r_b, G, \tilde{G}, \varepsilon_s, \varepsilon_{\rm mt}) = [Max(DistG(r_a, r_b, \varepsilon_s), DistMT(r_a, r_b, G, \tilde{G}, \varepsilon_{mt}))]^\beta \quad \text{with } \beta \in (0, 1] \tag{15}$$

Notice that we have applied the rectifier function as we have done in the Geographic-Temporal distance *DistGT* defined in formula (14).

5.7.1. Geographic-Modulo-Temporal epsilon

The *GT-DBSCAN* algorithm, when using the Geographic-Modulo-Temporal distance, is invoked as *GT-DBSCAN*(*DistGMT*, ε_s, ε_{mt},*MinPts*), where the values for the parameters ε_s and ε_{mt} are those used in Sections 5.3.1 and 5.5.1.

In this case, $Satisfy(DistGMT, r_a, r_b, \varepsilon_s, \varepsilon_{mt}, G, \tilde{G}) = (DistGMT(r_a, r_b, G, \tilde{G}, \varepsilon_s, \varepsilon_{mt}) < 1)$.

6. Case studies

In this section, we illustrate some experiments using the geographic temporal clustering defined in the previous section. For the sake of simplicity, we only consider one single information source s = *Twitter* and perform several explorations of *tweets*: Section 6.1 presents the data we have collected.

A proper evaluation of our approach would require the comparison of the results of the clustering with official data of the event under study. Since we do not always have such kind of information, we do a more qualitative analysis. First, in Section 6.2 we show the results of the application of our algorithms for different purposes on different collections of tweets dealing with several distinct topics, that we consider as main categories for the evaluation process.

Then, in Section 6.3 we perform a quantitative evaluation of the geo-temporal exploratory approach by setting up an experiment in which several user-driven geo-temporal explorations on the reports retrieved by distinct queries are performed.

In order to compute the validation measures we classified each report in the collection as belonging to both a *main* category and a *secondary* category, that is a subcategory in which each category is decomposed. As it will be explained in Section 6.1, the subcategories are chosen so as to test distinct “a priori” hypotheses on the geo-temporal characteristics of the event represented by the reports of a main category.

The evaluation experiment is designed so that four user queries are submitted to the collection of reports *R*(*Q*), each one aimed at selecting a sub-collection of the reports of one of the four main categories in which the data set is classified. Several geo-temporal clustering are run with distinct parameters and distance measures on each of the four sub-collections of selected reports. The quality of the results is evaluated by computing the percentage of clustered reports with respect to all the reports of the category, which is an indicator of the noise, the number of generated clusters, and the *F-measure* that is a balance of recall and precision of the clustering partition, representing the ability of the exploratory process to group all reports of a categories in clusters homogeneous with respect to the subcategories defined for the category. The sensitivity analysis of such validation measures is performed by varying the parameters and distance measure used by the clustering run and the best combination of parameters and distance is outlined for each of the four categories. The best combination is the one that provides the highest values of *%clustered* reports and *F-measure*. In particular, the identified distance measure that provided the best result on a given category of reports should be compliant with the “a priori” hypothesis on the geo-temporal characteristic of reports that brought to the definition of the subcategories.

Table 1
Tweets of the category “traffic jam” partitioned into the distinct subcategories defined by the hashtags in distinct languages in which “traffic jam” was translated.

Dutch	English	French	German	Greek	Italian	Japanese	Korean	Portuguese	Russian	Spanish	Thai	Turkish
225	521	20	105	4	92	15	19	83	134	369	161	118

Table 2
Number of collected tweets in several languages about floods, storms, inundations.

Chinese	Dutch	English	French	German	Indonesian	Italian	Korean	Portuguese	Russian	Spanish	Thai
3203	56	40,139	219	65	24,447	660	190	93	183	1350	1091

6.1. Data

Several queries have been formulated related to the following four events: *traffic jam*, the *US OPEN 2013 tennis tournament*, *floods, storms* and *inundations*, and the *soccer world cup 2014*. The queries were translated into hashtags and daily submitted to Twitter using the Twitter APIs[2] for a period of three months starting from July 2013 (except for the soccer world cup), and for two months starting from June 2014 (only for the soccer world cup). Totally, we collected 139348 tweets. The tweets collected by the queries related to one of the above four events are classified as belonging to one main category. Furthermore, the tweets of each category were also associated with a secondary category, i.e., subcategory that corresponds to the hashtag that retrieved them as described below.

As far as the traffic jam, the following query was specified:

$$q = ''\text{traffic jam}''$$

This query was translated, with the support of Google Translate APIs, into the several distinct hashtags {*#trafficjam*, *#traffico*, *#stau*, *#engarrafamento*, ...} (i.e., the term “traffic jam” was expressed in different languages: Dutch, English, French, German, Greek, Italian, Japanese, Korean, Portuguese, Russian, Spanish, Thai, and Turkish) that were separately submitted as queries to Twitter APIs. The number of retrieved tweets is shown in Table 1.

The subcategories of the traffic jam category are defined by the languages in Table 1. So a traffic jam report is classified as belonging to the subcategory according to the language in which it is written. Such definition of the subcategories is compliant with a geographic characterization of the reports.

We collected tweets related to the “US OPEN 2013” tennis tournament by formulating the query “*US open*” that was translated into the hashtag *#usopen* and submitted to Twitter APIs during the last week of the tournament, thus retrieving 5156 tweets. In this case, the tweets were associated with a match according to their dates: we considered the last 8 matches of the tournament. This choice of the subcategories is compliant with a temporal characterization of the reports.

Furthermore, we collected tweets related to floods storms and inundations formulating the query:

$$q = floods\, or\, storms\, or\, inundations.$$

The query was translated with the support of Google Translate APIs into the following hashtags in different languages ({*#flood*, *#inundation*, *#temporale*, *#storm*, ...}) and separately submitted to Twitter APIs for a period of two months starting from August 2013. The number of retrieves tweets is shown in Table 2.

In this case, there are twelve subcategories, identified by the hashtags in distinct languages, which is compliant with a geographic characterization of the reports.

Finally, we collected tweets related to the “soccer world cup 2014” by the query:

$$q = ''Brazil2014''\, or\, ''soccer\, world\, cup''$$

That was translated into the hashtags *#brasil2014* and *#worldcup* and submitted to Twitter APIs during June and July 2014, thus retrieving 60630 tweets. As subcategories, we classified the tweets according to the last eight matches of the tournament (i.e., starting from the quarterfinals) which is compliant with a temporal characterization.

6.2. Clustering single categories of items using distinct distance measures

In order to exemplify the utility of using the different distance measures, in this section we only consider the tweets related to the traffic jam and those related to the tennis tournament. For each distance, we use the category that is more suitable for showing the performance of the clustering algorithm.

The validation of the approach by considering all the collected tweets is discussed in Section 6.3.

[2] https://dev.twitter.com/docs/api

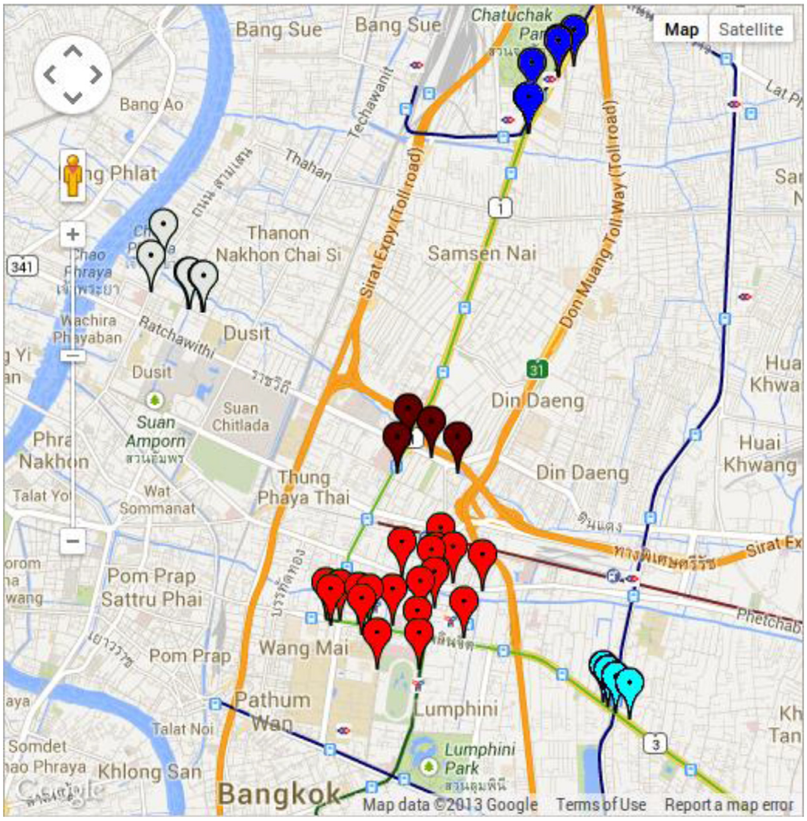


Fig. 3. Results of the GT-DBSCAN clustering using the Geographic distance *DistG* in formula (10) – "Traffic jam" in Bangkok.

6.2.1. Geographic clustering

Geographic clustering by GT-DBSCAN can be very useful to identify local regions interested by long term traffic jams, where most probably several tweets are originated from the same congested area.

Fig. 3 shows the results of the application of GT-DBSCAN for the Bangkok area to the tweets related to the traffic jam.

The algorithm was run as follows

$$\mathrm{GT-DBSCAN}(\mathrm{DistG}, 0.5\mathrm{km}, \propto, -, -, 4) \tag{16}$$

This means that *MinPts* = 4 and the maximum geographic distance is $\varepsilon_s = 0.5$ km; different colors identify different clusters that have been generated. We can see that there is an area with a big cluster in the middle of the city (plausibly, a traffic congested area) and small clusters on the streets entering this area.

Geographic clustering can also be used to identify regions in which people are following the same event. For example, the geographic clustering of the tweets related to the soccer world cup permits to identify the nations more interested in the competition.

Fig. 4 shows the results of the application of GT-DBSCAN to the tweets related to the soccer world cup all over the world.

The algorithm was run as follows

$$\mathrm{GT-DBSCAN}(\mathrm{DistG}, 20\mathrm{km}, \propto, -, -, 67) \tag{17}$$

This means that *MinPts* = 67 and the maximum geographic distance is $\varepsilon_s = 20$ km.

We can see that the world cup has been followed mainly in Europe and in America. Moreover, as expected, the majority of tweets came from the biggest cities in the world.

6.2.2. Temporal clustering

Temporal clustering by GT-DBSCAN is particular useful for identifying the date and the hours of global events that may occur all over the globe. In order to show the temporal clustering, we consider the tweets about the US OPEN 2013; indeed, people tend to send tweets about a sport event while the event is happening and is broadcasted on TV worldwide. Fig. 5 shows the temporal clustering of the tweets about the US OPEN 2013 tennis tournament. This has been obtained by running GT-DBSCAN with the following settings:

$$\mathrm{GT-DBSCAN}(\mathrm{DistT}, \propto, 10\ \mathrm{min}, \mathrm{min}, -, 90) \tag{18}$$

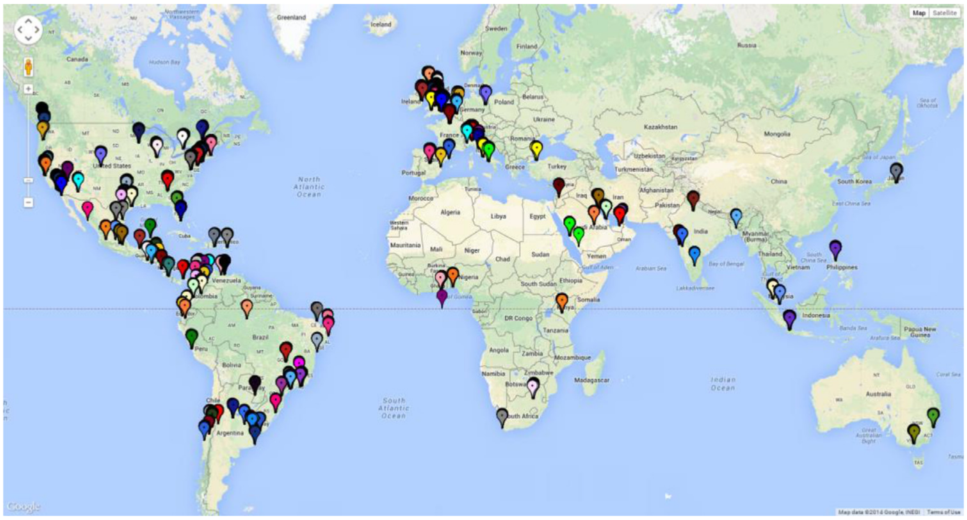


Fig. 4. Results of the GT-DBSCAN clustering using the Geographic distance *DistG* in formula (10) – “Soccer World cup”.

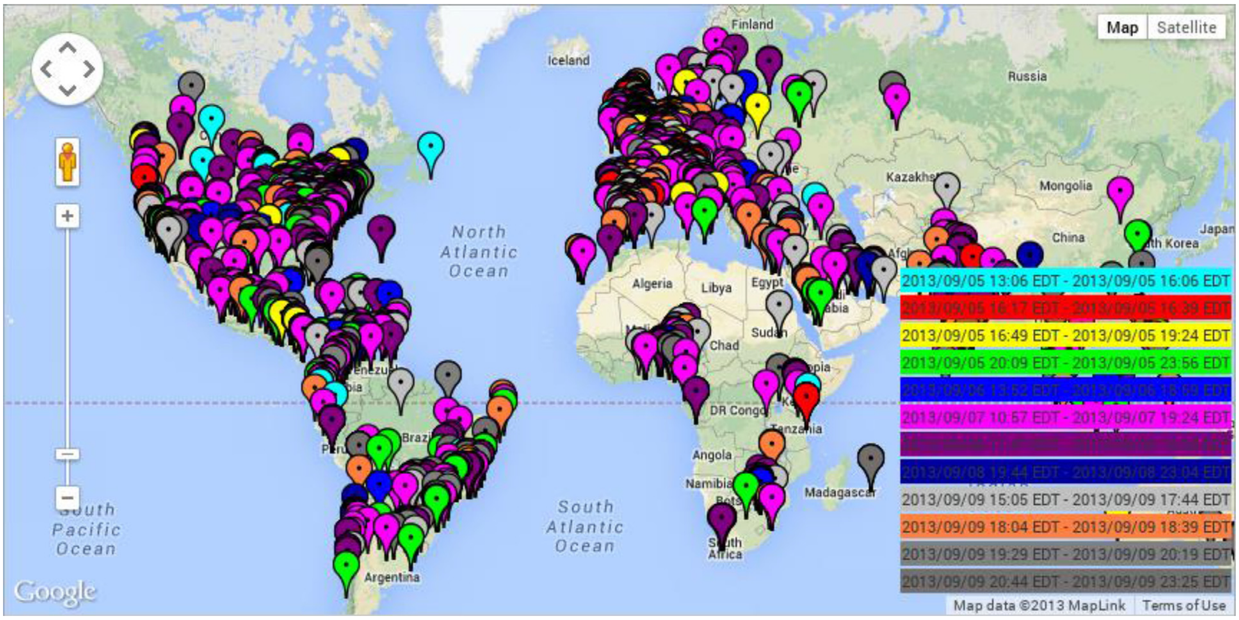

Map
Satellite
Finland
Iceland
Russia
Kazakhstan
Mongolia
China
Japan
South Korea
North Atlantic Ocean
Algeria
Libya
Egypt
Sudan
Chad
DR Congo
Tanzania
Angola
Namibia
Madagascar
South Africa
Peru
Brazil
Argentina
South Pacific Ocean
South Atlantic Ocean
2013/09/05 13:06 EDT - 2013/09/05 16:06 EDT
2013/09/05 16:17 EDT - 2013/09/05 16:39 EDT
2013/09/05 16:49 EDT - 2013/09/05 19:24 EDT
2013/09/05 20:09 EDT - 2013/09/05 23:56 EDT
2013/09/06 13:52 EDT - 2013/09/06 18:59 EDT
2013/09/07 10:57 EDT - 2013/09/07 19:24 EDT
2013/09/08 19:44 EDT - 2013/09/08 23:04 EDT
2013/09/09 15:05 EDT - 2013/09/09 17:44 EDT
2013/09/09 18:04 EDT - 2013/09/09 18:39 EDT
2013/09/09 19:29 EDT - 2013/09/09 20:19 EDT
2013/09/09 20:44 EDT - 2013/09/09 23:25 EDT
Google
Map data ©2013 MapLink
Terms of Use

Fig. 5. Results of the GT-DBSCAN clustering using the Temporal distance *DistT* in formula (11) – Tweets about the US Open 2013.

which means that $MinPts = 90$ and the temporal distance for the clustering is $\varepsilon_t = 10$ min.

We have cross compared each cluster time range with the times of the main matches of the tournament and realized that each cluster is associated with a main event. In fact, it can be noticed that the 12 identified temporal clusters group tweets created during one of the last five matches of the tournament, specifically:

- 5 Sept 2013, 13:06 EDT –23:56 EDT: Men’s Quarterfinals
- 6 Sept 2013, 13:52 EDT–18:59 EDT: Women’s Semifinals
- 7 Sept 2013, 10:57 EDT–19:24 EDT: Men’s Semifinals
- 8 Sept 2013, 11:47 EDT–23:04 EDT: Women’s Final
- 9 Sept 2013, 15:05 EDT–23:25 EDT: Men’s Final

We have also realized that the sum of the cardinality of the clusters with timestamp “9 September 2013”, corresponding to the Men’s Final match, is the greatest, thus revealing that this match has been the most popular at a global scale.

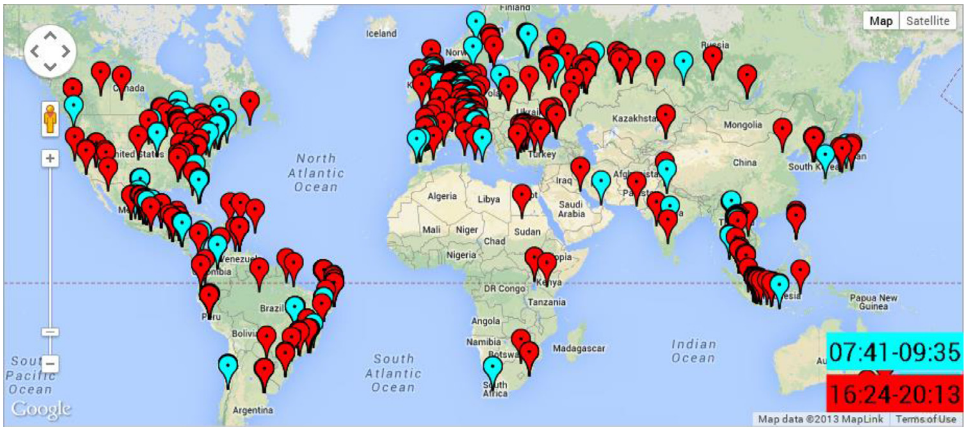


Fig. 6. Results of the GT-DBSCAN clustering using the Modulo-Temporal distance *DistMT* in formula (12) – Tweets about Traffic jam all over the world are clustered into two main time intervals of the day.

6.2.3. Modulo-temporal clustering

Modulo-temporal clustering by GT-DBSCAN is useful for identifying recurring events that may occur both globally and locally. An example of recurring event could be traffic jam; indeed, if we want to identify, at a global scale, which are the time intervals of the day in which traffic jams occur more frequently on Earth, we can run GT-DBSCAN using the modulo-temporal clustering distance, by disregarding the geographic location where the tweets have been created. We run the GT-DBSCAN on the collection of the tweets related to the traffic with the following settings:

$$\mathrm{GT-DBSCAN}(\mathrm{DistMT}, \propto, 2\mathrm{min}, \mathrm{min}, \mathrm{day}, 100) \tag{19}$$

This means that the temporal distance is $\varepsilon_{mt} = 2$ min, $\tilde{G}$= *day* is considered as the period of interest, and $MinPts = 100$.

Fig. 6 shows the results of modulo-temporal clustering of all the tweets about traffic jam, submitted all over the world.

Two temporal intervals of the day have been identified by two clusters, one in the morning (7:41–9:35) and one in the late afternoon (16:24–20:13); as expected, these are the temporal intervals in which there are more traffic jams all over the world (when people go to or return from the work place). We can notice that there are much more tweets in the afternoon interval. This may be due to the fact that the temporal interval of the afternoon cluster is wider than the temporal interval of the morning cluster. However, we do not know if there is actually more congestion in the afternoon, or if people tend to complain more in the afternoon, maybe because they are tired of their working day and they want to reach home quickly. Notice that the unpredictability of human behavior is a bias of our approach [15].

Fig. 7 shows the results of GT-DBSCAN using the modulo-temporal distance in the Bangkok area, with a different setting with respect to the previous run.

In this case, the setting is:

$$\mathrm{GT-DBSCAN}(\mathrm{DistMT}, \propto, 12\ \mathrm{min}, \mathrm{min}, \mathrm{day}, 10) \tag{20}$$

It demands less points than before to create a core point ($MinPts = 10$), and the temporal distance between tweets timestamps is larger than in the previous run ($\varepsilon_{mt} = 12$ min), while the period is still the day. We discovered two macro-clusters, one in the interval 7:01–8:36 (actually composed of two clusters close in time) and the other in the interval 17:00–21:16. We can see that the discovered temporal intervals, although similar to those discovered for the whole world, are more related to the local habits in Bangkok: for example, the afternoon cluster is shifted ahead of about 1 h.

6.2.4. Geo-temporal clustering

Geo-Temporal clustering by GT-DBSCAN clustering is useful for analyzing events in relation with the same geographic areas. An example of such events may be tweets related to the traffic jam: indeed, different people stuck in the same congested road may send tweets related to the congestion. We applied GT-DBSCAN to the tweets related to the traffic jam by using the *DistGT* distance in formula (14) with the following settings of the input parameters:

$$\mathrm{GT-DBSCAN}(\mathrm{DistGT}, 2\ \mathrm{km}, 20\ \mathrm{min}, -, -, 2) \tag{21}$$

which corresponds to $MinPts = 2$, $\varepsilon_s = 2$ km, and $\varepsilon_t = 20$ min. Fig. 8 shows one geo-temporal cluster found in the city of Gurgaon in India (August 11, 2013): the same Twitter user has sent three tweets while stuck in the traffic congestion.

Fig. 9, instead, shows a cluster found in Jakarta (July 13, 2013) where two different users are complaining about the same congestion.

Finally, Fig. 10 shows two clusters found in Lima (August 22, 2013 and August 23, 2013).

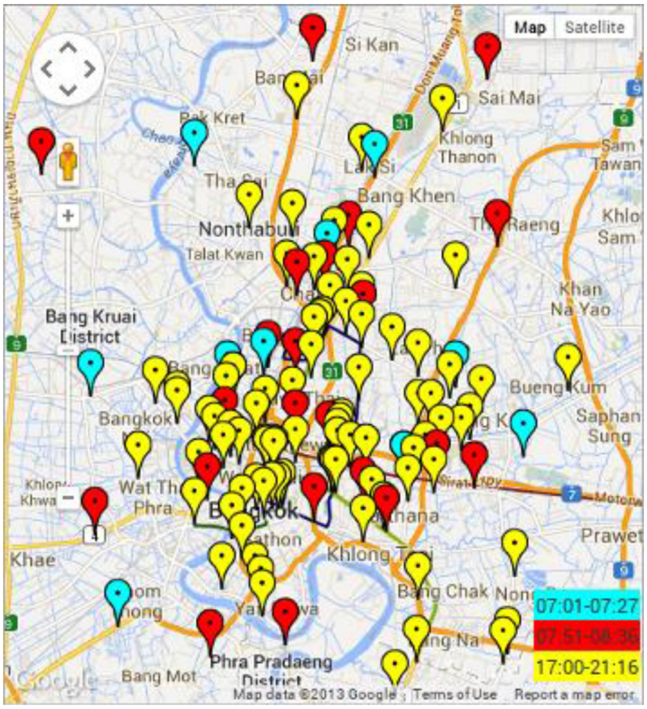


Fig. 7. Results of the GT-DBSCAN clustering using the Modulo-Temporal distance *DistMT* in formula (12) – Tweets about Traffic jam in Bangkok.

6.2.5. Geo-modulo-temporal clustering

Geo-modulo-temporal clustering by GT-DBSCAN (i.e., using *DistGMT* distance in formula (15)) is useful for identifying the periods of time when events regularly occur in relation with the same area. We consider the tweets related to the traffic jams which are events that may have a periodicity in specific regions; for example, it may happen that a particular road is always congested in a specific moment of the day.

Fig. 11 shows the results of GT-DBSCAN using the *DistGMT* distance on the tweets related to the traffic jam in the Bangkok area with the following setting of the parameters:

$$\mathrm{GT-DBSCAN(DistGMT, 1km, 20min, min, day, 3)} \tag{22}$$

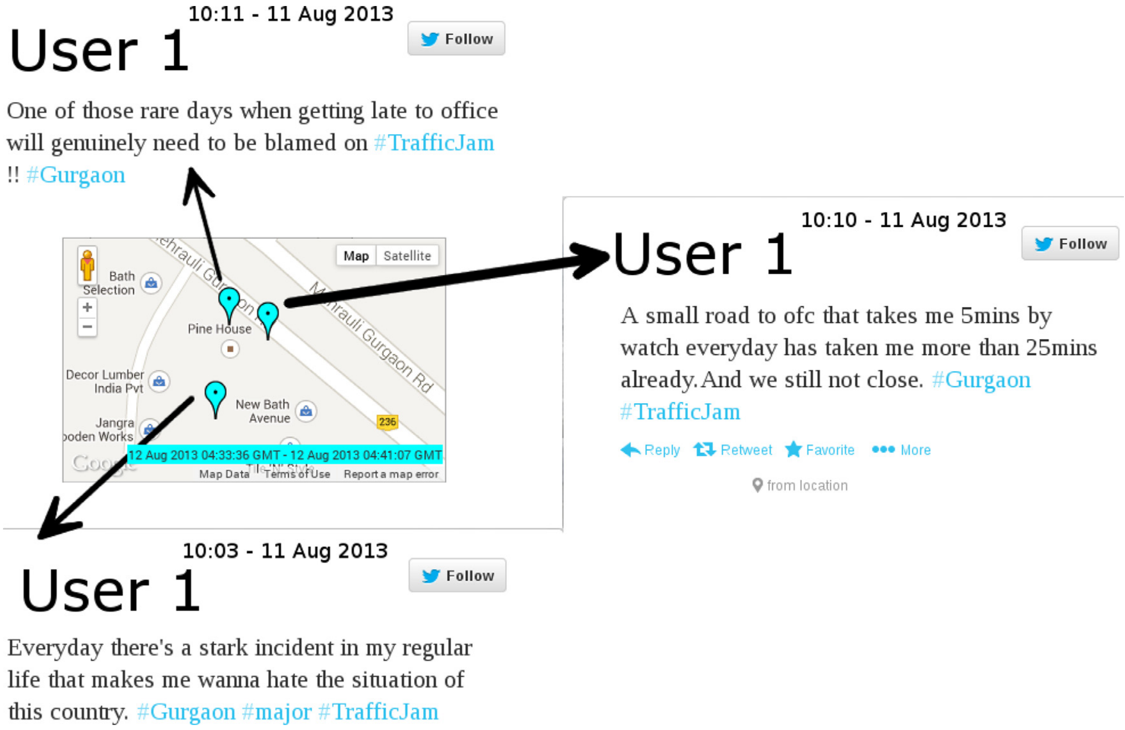
which means $MinPts = 3$, $\varepsilon_s = 1$ km,$\varepsilon_{mt} = 20$ min, and $modulo = G_d =$ day.

We can see the congested areas in relation with the interval of time when the congestions occur. Some clusters slightly correspond to those discovered in the same area by running GT-DBSCAN with the Geographic distance (see Fig. 3): nevertheless, in this analysis the clusters have been filtered by the time constraint, and thus they are smaller. For example, the big cluster shown in Fig. 3 is reduced in size in Fig. 11, since only tweets belonging to the temporal interval 15:08–15:44 have been kept. Some other clusters in Fig. 3, instead, have not been identified when considering also the temporal constraint because they do not satisfy it. The two clusters related to the time intervals 17:38–18:09 and 18:06–18:26 in Fig. 11 have not been identified in Fig. 3 since the required number of reports per cluster was too high.

6.3. Validation experiment

Last but not least a quantitative evaluation of the proposed user-driven exploratory approach, by considering all the 139,348 collected tweets in the crawling phase, has been performed.

To this end, we designed an original experiment in which a user first selects a subset of the tweets about an event of interest, i.e., a sub-collection (the tweets in one of the four main categories). Then, he/she applies some geo-temporal explorations of the retrieved sub-collection by specifying distinct distance measures and parameters for the GT-DBSCAN clustering. The results yielded by each run of the clustering on the same sub-collection are compared in order to assess which is the combination of distance measure and parameters that yield the best values of the validation measures. As validation measures we computed the number of automatically generated clusters *(#clusters*), the percentage of clustered reports, that is the inverse function of the found noise (*%clustered*), and the *F-measure*, that is a combination of recall and precision of the clustering partition. Precision and Recall were calculated with respect to the subcategories, in order to estimate how good is the exploratory process in clustering all reports of a category into homogeneous clusters with respect to the subcategories.



10:11 - 11 Aug 2013

User 1

One of those rare days when getting late to office will genuinely need to be blamed on #TrafficJam !! #Gurgaon

10:10 - 11 Aug 2013

User 1

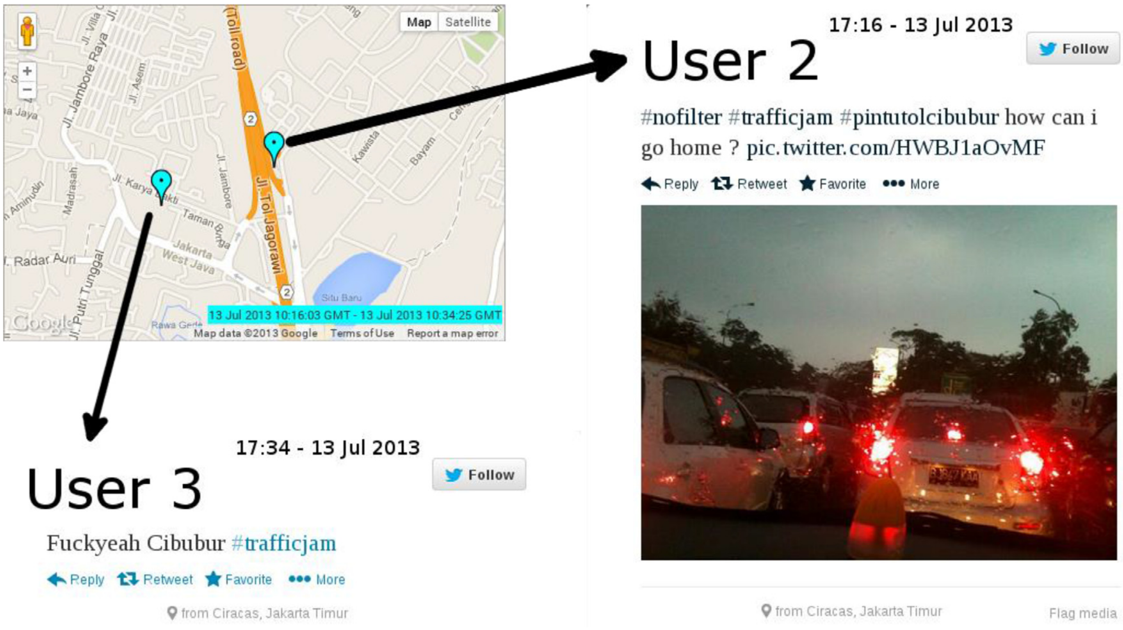
A small road to ofc that takes me 5mins by watch everyday has taken me more than 25mins already. And we still not close. #Gurgaon #TrafficJam

10:03 - 11 Aug 2013

User 1

Everyday there's a stark incident in my regular life that makes me wanna hate the situation of this country. #Gurgaon #major #TrafficJam

Fig. 8. Results of the GT-DBSCAN clustering using the Geo-Temporal distance *DistGT* in formula (14) – Traffic jam in Gurgaon, India.



17:16 - 13 Jul 2013

User 2

#nofilter #trafficjam #pintutolcibubur how can i go home ? pic.twitter.com/HWBJ1aOvMF

17:34 - 13 Jul 2013

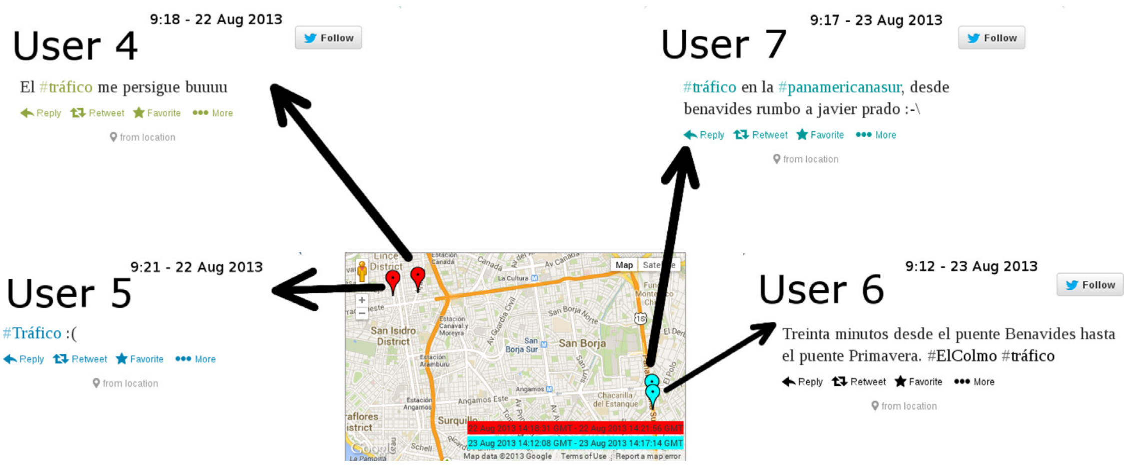
User 3

Fuckyeah Cibubur #trafficjam

Fig. 9. Results of the GT-DBSCAN clustering using the Geo-Temporal distance *DistGT* in formula (14) – Traffic jam in Jakarta, Indonesia.

The idea is that the clustering run with best performance of the validation measures should reveal what is the most proper geo-temporal nature of the event, which should be compliant with the geo-temporal characterization provided by the chosen subcategories.

Given, as a result of the clustering process on a sub-collection, k clusters $C = \{c_1, \ldots, c_k\}$ and considering that a sub-collection of tweets is partitioned into j subcategories $g_1, \ldots, g_j$, the *F- measure* is computed by applying the following



User 4 — 9:18 - 22 Aug 2013

El #tráfico me persigue buuuu

User 7 — 9:17 - 23 Aug 2013

#tráfico en la #panamericanasur, desde benavides rumbo a javier prado :-\

User 5 — 9:21 - 22 Aug 2013

#Tráfico :(

User 6 — 9:12 - 23 Aug 2013

Treinta minutos desde el puente Benavides hasta el puente Primavera. #ElColmo #tráfico

Follow
Reply
Retweet
Favorite
More
from location
Map
Satellite
Lince District
San Isidro District
San Borja
Surquillo
Miraflores District
22 Aug 2013 14:18:31 GMT - 22 Aug 2013 14:21:56 GMT
23 Aug 2013 14:12:08 GMT - 23 Aug 2013 14:17:14 GMT
Map data ©2013 Google Terms of Use Report a map error

Fig. 10. Results of the GT-DBSCAN clustering using the Geo-Temporal distance *DistGT* in formula (14) – Traffic jam in Lima, Peru.

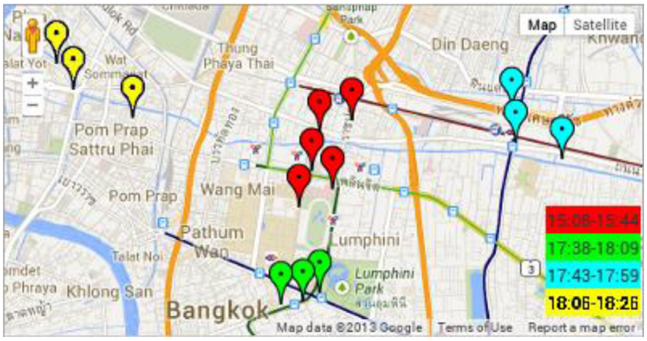


Fig. 11. Results of the GT-DBSCAN clustering using the Geo-Modulo-Temporal distance *DistGMT* in formula (15) – Traffic jam in Bangkok.

definition:

$$\text{F} - \text{measure}(C) = 1 - \left(\frac{\alpha}{\text{Precision}(C)} + \frac{1-\alpha}{\text{Recall}(C)}\right) \text{ with } \alpha \in [0,1] \tag{23}$$

In the evaluation reported in the paper, we set $\alpha = 0.5$ so as to take into account a balance of Recall and Precision.

As far as the *recall* of a clustering partition, it is defined as the average of the recall of the partition for all the subcategories, i.e., the greatest fraction of elements of a cluster belonging to a subcategory with respect to the subcategory cardinality ($|a|$ denotes the cardinality of the set a), while the *precision* of a clustering partition is defined as the average of the precision of the partition for all the subcategories, i.e., the greatest fraction of elements of a cluster belonging to a subcategory with respect to the cluster cardinality:

$$\text{Recall}(C) = \frac{\sum_{h=1}^{j} recall(g_h)}{j} \qquad \text{with} \quad recall(g_h) = \max_{i=1,\dots,k} \frac{(|c_i \cap g_h|)}{|g_h|} \tag{24}$$

$$\text{Precision}(C) = \frac{\sum_{h=1}^{j} precision(g_h)}{j} \qquad \text{with} \quad precision(g_h) = \max_{i=1,\dots,k} \frac{(|c_i \cap g_h|)}{|c_i|} \tag{25}$$

A sensitivity analysis of the GT-DBSCAN clustering was studied by varying both the parameters defining the local density constraint (i.e., *minPts*, minimum number of reports), ε_s and ε_t (i.e., the maximum spatial and temporal distances), and period G when using *DistMT* and *DistGMT*. Fig. 12a–e illustrates, for the sub-collection of traffic jam reports, the variation of *F-measure*, of the number of generated clusters *(#clusters)*, and of the percentage of clustered reports (*%clustered*), by using distinct distance measures and distinct values of the parameters. *F-measure* is computed with respect to the 13 subcategories of the traffic jam category. It can be observed that the validation measures tend to decrease in almost all the cases by increasing *minPts*. The *F-measure* is always below 0.25 when using both the aperiodic and periodic time distance *DistT* and *DistMT*, and the geo-time distance *DistGT*, while it increases to 0.69 when using both the geographic distance *DistG* and the periodic geo-temporal distance *DistGMT*.

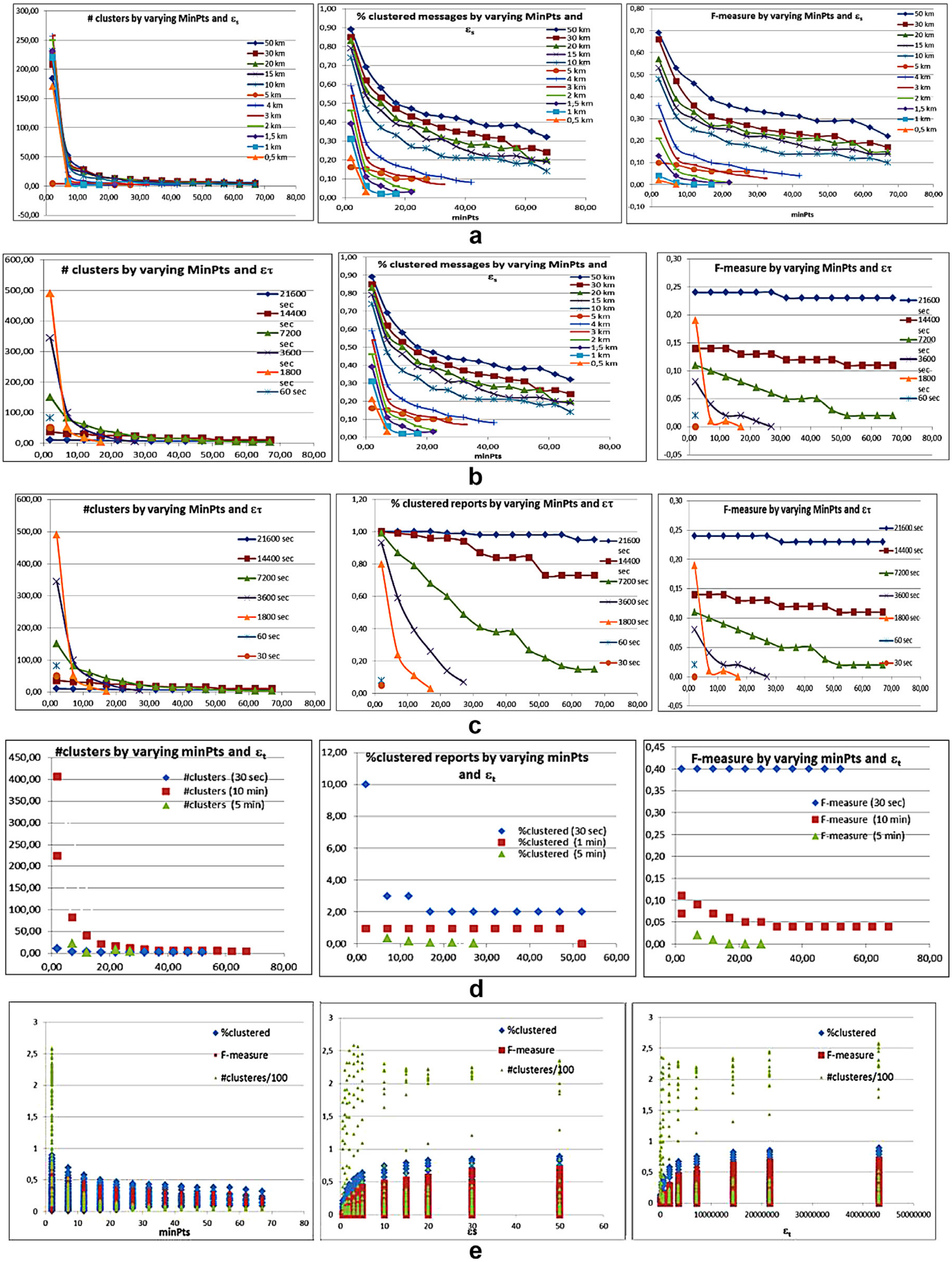


Fig. 12. (a) Traffic jams: sensitivity analysis of GT-DBSCAN using the geographic distance *DistG* in formula (10). (b) Traffic jams: sensitivity analysis of GT-DBSCAN using the time distance *DistT* in formula (11). (c) Traffic jams: sensitivity analysis of GT-DBSCAN using geo-temporal distance *DistGT* in formula (14). (d) Traffic jams: sensitivity analysis of GT-DBSCAN using periodic modulo-temporal distance *DistMT* in formula (12). (e) Traffic jams: sensitivity analysis of GT-DBSCAN using geo-modulo temporal distance *DistGMT* in formula (15).

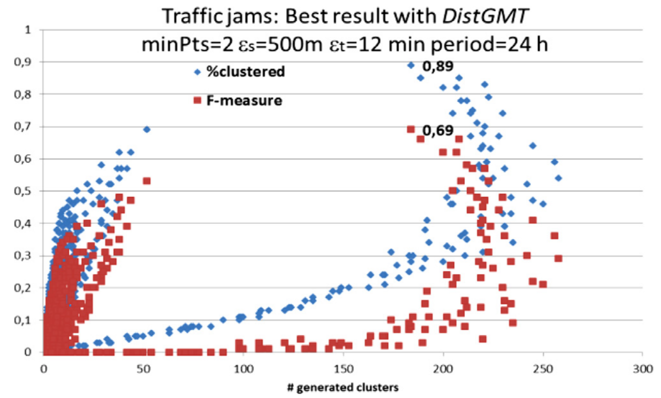


Fig. 13. Traffic jams: *F-measure* and *%clusterized* reports as *#clusters* (the number of generated clusters) varies when using the distance *DistGMT* in formula (15).

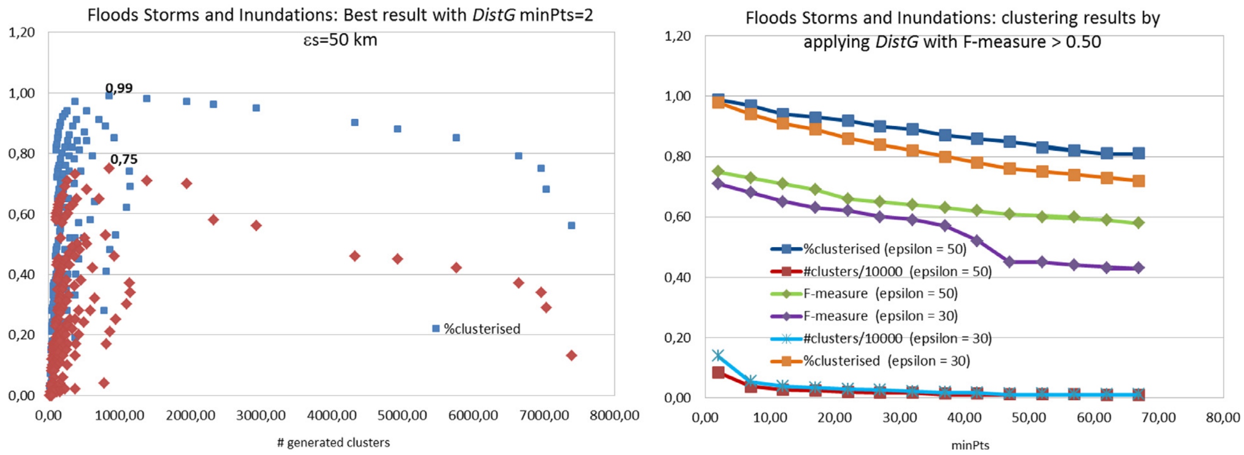


Fig. 14. Floods, Storms and Inundations clustered by using the distance *DistG* in formula (10): (*a*) *F-measure* and *%clusterized* reports as the number of generated clusters varies, (b) variation of the number of generated clusters (*#clusters*), %clustered reports (*%clustered*) and *F-measure* by increasing *minPts* and the other parameters.

Specifically, in Fig. 13, it can be observed that the best values of *F-measure* and *%clustered* reports are obtained with *DistGMT*, $minPts = 2$ $\varepsilon_s = 500$ m,$\varepsilon_t = 12$ min, and period = 24 h, assuming the values 0.69 and 89% respectively.

This result reveals that the reports about the collected traffic jams have a periodic and geographic characterization, since they have a daily frequency with locally dense geographic distribution (as outlined by *DistGMT*), which is compliant with the fact that the used subcategories classify the reports according to their language.

Fig. 14a and b reports the best results *%clusterized* = 99% and *F-measure* = 0.75 obtained by applying the geographic distance *DistG* on the sub-collection of Floods, storms and inundations, comprising 12 subcategories. In this case, the results with *DistGMT* are worse than those obtained by using *DistG*, so revealing that the collected reports do not have a periodic characterization, since they essentially have a geographic characterization, which is compliant with the choice of the subcategories. Also in this case it can be seen in Fig. 14b that, by increasing the parameter *minPts* and by decreasing the parameter ε_s which defines the maximum distance, both the *%clusterized* and *F-measure* decrease.

Fig. 15a and b reports the best results *%clusterized* = 99% and *F-measure* = 0.54 obtained by applying the time distance *DistT* on the sub-collection of the US Open 2013 tennis tournament, comprising the 8 subcategories of its main matches. It can be observed that both *%clusterized* = 99% and *F-measure* are stable by varying *minPts* from 2 to 67 and the temporal maximum distance ε_t from 2 to 12 h, while they both decrease for values of the time distance below 2 h. This is also compliant with the choice of the subcategories that correspond with the tennis matches.

Fig. 16a and b reports the best results *%clusterized* = 84% and *F-measure* = 0.73 obtained by applying the time distance *DistT* on the sub-collection of the Soccer World Cup 2014 comprising the 8 subcategories of its main matches. This is also compliant with the choice of the subcategories that correspond with the main matches.

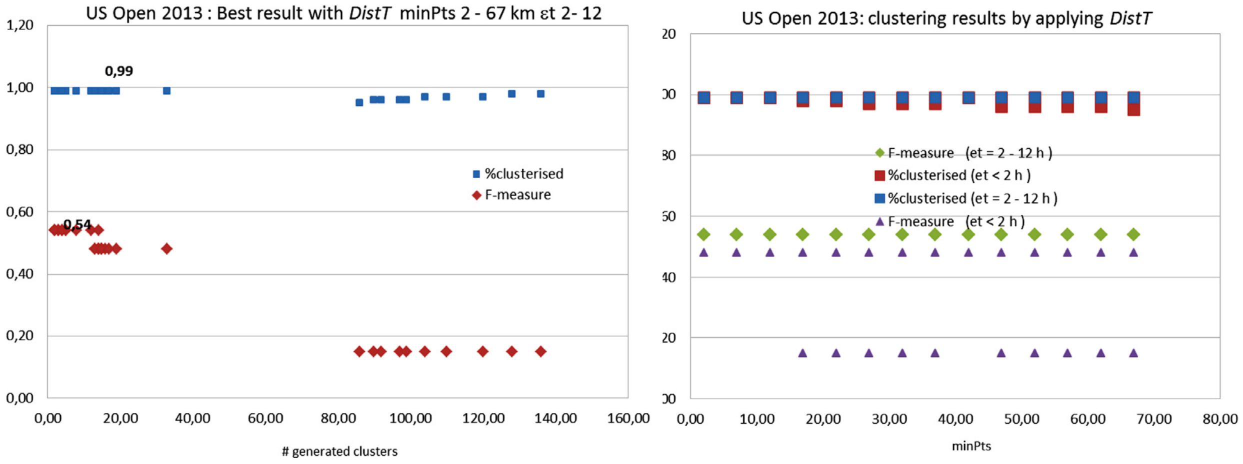


Fig. 15. US Open 2013 clustered by using the distance *DistT* in formula (11), (a) *F-measure* and *%clusterized* reports as the number of generated clusters varies, (b) variation of the number of generated clusters (*#clusters*), %clustered reports (*%clustered*) and *F-measure* by increasing *minPts* and the other parameters.

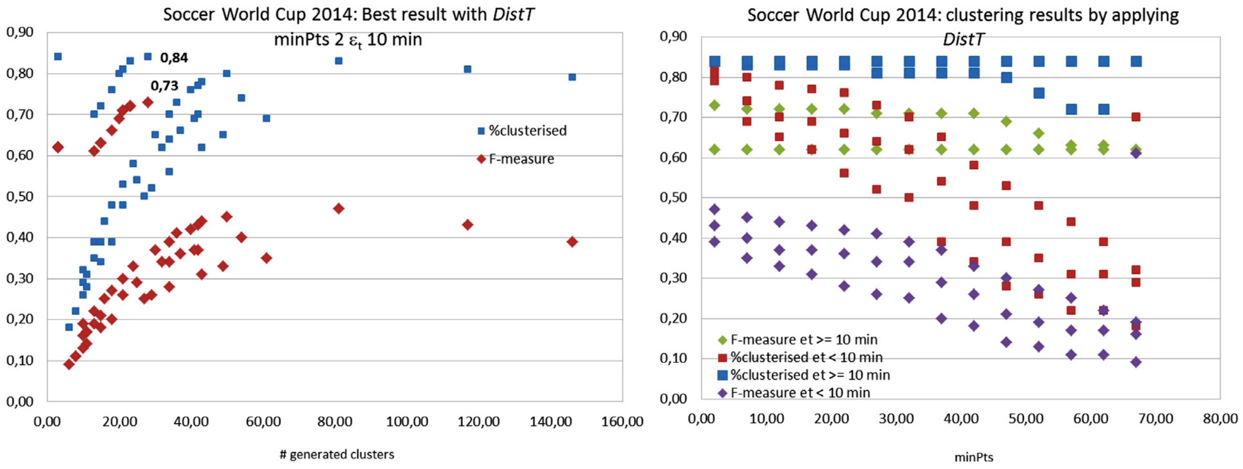


Fig. 16. Soccer World Cup 2014 clustered by using the distance *DistT* in formula (11), (a) *F-measure* and *%clusterized* reports as the number of generated clusters varies, (b) variation of the number of generated clusters (*#clusters*), %clustered reports (*%clustered*) and *F-measure* by increasing *minPts* and the other parameters.

7. Conclusions

We have proposed a user-driven approach to perform a flexible geo-temporal analysis of information about events of interest in social network contexts. The procedure has been implemented for analyzing Twitter messages, collected by a focused crawler by querying the source of messages over a period of time, and subsequently by allowing a user to query the selected messages and then to apply an original extension of the DBSCAN clustering algorithm to identify geo-temporal clusters of messages. The approach can be applied to any message which contains metadata related to its geographic location and timestamp, so it is well suited to analyze geo-referenced messages.

One main advantage of the proposal with respect to other approaches performing spatio-temporal analysis of social networks is its flexibility to adapt the exploration to user's needs. This is done by specifying a query to filter messages with interesting contents and then by selecting a specific distance measure with desired parameters defining the local density of messages in space and time to drive the grouping. This allows identifying clusters of information items with different geographic, temporal, and geo-temporal characteristics. Specifically, the modulo-temporal and the geo-modulo-temporal distances have been defined to identify clusters of information items whose timestamps are characterized by a desired periodicity. As far as we know, there is no clustering algorithm detecting periodic events.

One main limitation of the proposal is the fact that the user needs to have some "a priori" hypothesis on the spatial and temporal characteristics of the events in order to express the exploratory criteria that drive the analysis that is regarded as a verification of the hypothesis. Nevertheless, when one is completely unaware of the characteristic of an event, an optimization process could be defined, aimed at identifying which distance measure is the most appropriate for a given collection of messages in terms of internal metrics of the obtained partitions.

As future work, we plan to validate the approach with official data of the event under analysis; moreover, we plan to investigate techniques for (semi-)automatically setting the best parameters of GT-DBSCAN.

Acknowledgments

This work has been carried out within the partial support of the project SISTEMATI “Geomatics Supporting Environmental, Technological and Infrastructural Disaster Management”, funded by the Italian Ministry of Research jointly with Regione Lombardia, and by Charles University research funds PRVOUK.

References

[1] P. Arcaini, G. Bordogna, S. Sterlacchini, Flexible querying of volunteered geographic information for risk management, in: Proceedings of the 8th International Conference of the European Society for Fuzzy Logic and Technologies (EUSFLAT 2013), Milan, Italy, September 11-13, 2013.

[2] S. Ardon, A. Bagchi, A. Mahanti, A. Ruhela, A. Seth, R.M. Tripathy, S Triukose, Spatio-temporal and events based analysis of topic popularity in twitter, in: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management (CIKM ’13), New York, NY, USA, ACM, 2013, pp. 219–228.

[3] F. Bießmann, J.M. Papaioannou, A. Harth, M.L. Jugel, K.R. Muller, M. Braun, Quantifiying spatio temporal dynamics of tweeters replies to news feeds, in: Proceedings of the 2012 IEEE International Workshop on Machine Learning for Signal Processing, 2012, pp. 23–26.

[4] D. Birant, A. Kut, ST-DBSCAN: An algorithm for clustering spatial-temporal data, Data Knowl. Eng 60 (1) (January 2007) 208–221.

[5] J. Bollen, H. Mao, A Pepe, Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena, in: Proceedings of ICWSM, 2011.

[6] G. Bordogna, F. Bucci, P. Carrara, M. Pepe, A. Rampini, Flexible querying of imperfect temporal metadata in spatial data infrastructures, Advanced Database Query Systems: Techniques, Applications and Technologies, IGI Global, 2011, pp. 140–159.

[7] G. Bordogna, P. Carrara, L. Criscuolo, M. Pepe, A. Rampini, A linguistic decision making approach to assess the quality of volunteer geographic information for citizen science, Inf. Sci. 258 (2014) 312–327.

[8] J. Chae, D. Thom, H. Bosch, Y. Jang, R Maciejewski, Spatiotemporal social media analytics for abnormal event detection and examination using seasonal-trend decomposition, in: Proceedings of the Conference on Visual Analytics Science and Technology (VAST), 2012, IEEE, 14-19 Oct. 2012, pp. 143–152.

[9] S. Chandra, L. Khan, F.B Muhaya, Estimating twitter user location using social interactions – a content based approach, in: Proceedings of 2011 IEEE International Conference on Privacy, Security, Risk, and Trust, and IEEE International Conference on Social Computing, 2011, pp. 838–843.

[10] T. Cheng, T. Wicks, Event detection using Twitter: A spatio-temporal approach, online PLOS one, 3/6/2014.

[11] M. Ester, H.P. Kriegel, J. Sander, and X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96).*AAAI Press, 226–231.

[12] M.F Goodchild, Citizens as voluntary sensors: spatial data infrastructure in the World of Web 2.0, Int. J. Spat. Data Infrastruct. Res. 2 (2007) 24–32.

[13] M.F. Goodchild, L. Li, Assuring the quality of volunteered geographic information, Spat. Stat. 1 (2012) 110–120.

[14] E. Hand, Citizen science: People power, Nature 466 (7307) (2010) 685–687.

[15] K.Y. Kamath, J. Caverlee, K. Lee, Z. Cheng, Spatio-temporal dynamics of online memes: a study of geo-tagged tweets, in: Proceedings of the ACM WWW 2013, Rio de Janeiro, Brazil, 2013.

[16] B. Krishnamurthy, P. Gill, M. Arlitt, A few chirps about twitter, in: Proceedings of the ACM Sigcomm Workshop on Social Networks, WOSN’08, 2008.

[17] H. Kwak, C. Lee, H. Park, S. Moon, What is twitter, a social network or a news media? in: Proceedings of the 19th International Conference on World wide Web, WWW ’10, ACM, 2010, pp. 591–600.

[18] C.H. Lee, Mining spatio-temporal information on microblogging streams using a density-based online clustering method, Expert Syst. Appl. 39 (10) (August 2012) 9623–9964.

[19] J. Leskovec, L. Backstrom, J. Kleinberg, Meme-tracking and the dynamics of the news cycle, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’09, ACM, 2009, pp. 497–506.

[20] Q. Liu, M. Deng, Bi J., W. Yang, A novel method for discovering spatio- temporal clusters of different sizes, shapes, and densities in the presence of noise, Int. J. Digit. Earth (2012) 1–20.

[21] M. Nagarajan, K. Gomadam, A.P. Sheth, A. Ranabahu, R. Mutharaju, and A. Jadhav, Spatio-temporal-thematic analysis of citizen sensor data: challenges and experiences, in: *Proceedings of WISE ’09*, Springer-Verlag, Berlin, Heidelberg, 539–553.

[22] B. O’Connor, R. Balasubramanyan, B.R. Routledge, N.A. Smith, From tweets to polls: linking text sentiment to public opinion time series, in: Proceedings of the International AAAI Conference on eblogs and Social Media, 2010.

[23] O. Okolloh, Ushahidi, or ’testimony’: Web 2.0 tools for crowdsourcing crisis information, Particip. Learn. Action 59 (1) (2009) 65–70.

[24] R. Oliveira, M.Y. Santos, J.M. Pires, 4D+ SNN: A spatio-temporal density-based clustering approach with 4D similarity, in: Proceedings of the IEEE 13th International Conference on Data Mining Workshops (ICDMW), IEEE, 2013, pp. 1045–1052.

[25] Y. Raimond, S. Abdallah, The event ontology, 2007. http://motools.sourceforge.net/event/event.html (accessed 06.10.15).

[26] D.M. Romero, B. Meeder, J. Kleinberg, 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, WWW ’11, Hyderabad, India, 2011, pp. 695–704.

[27] B. Saaid, D.A. Simovici, C. Zara, The impact of triangular inequality violations on medoid-based clustering, in: Proceedings of the 19th International Symposium ISMIS 2011, Foundation of Intelligent Systems, Springer Verlag, 2011, pp. 280–289.

[28] T. Sakaki, M. Okazaki, Y. Matsuo, Earthquake shakes twitter users: real-time event detection by social sensors, earthquake shakes twitter users: real-time event detection by social sensors, in: Proceedings of the 19th International Conference on World Wide Web, WWW ’10, New York, NY, USA, ACM, 2010, pp. 851–860.

[29] R.W. Sinnott, Virtues of the Haversine, Sky Telesc. 68 (2) (1984) 159.

[30] D. Thom, H. Bosch, S. Koch, M. Woerner, T. Ertl, Spatio temporal anomaly detection through visual analysis of geolocated twitter messages, in: Proceedings of the IEEE Pacic Visualization Symposium (PacicVis), 2012.

[31] K. Watanabe, M. Ochi, M. Okabe, R. Onai, Jasmine: a real-time local-event detection system based on geolocation information propagated to microblogs, in: Proceedings of the 20th ACM international Conference on Information and Knowledge Management, CIKM ’11, 2011, pp. 2541–2544.

[32] S. Wu, C. Tan, J. Kleinberg, M. Macy, Does bad news go away faster? in: Proceedings of the 5th International AAAI Conference on Weblogs and Social Media, ICWSM ’11, 2011, pp. 646–649.

[33] J. Yang, J. Leskovec, Patterns of temporal variation in online media, in: Proceedings of the 4th ACM International Conference on Web Search and Data Mining, WSDM ’11, Hong Kong, China, ACM, 2011, pp. 177–186.

[34] S. Yardi, D. Boyd, Tweeting from the town square: Measuring geographic local networks, in: Proceedings of the 4th International AAAI Conference of Weblogs and Social Media, The AAAI Press, 2010.

[35] H. Zhang, M. Korayem, E. You, D.J. Crandall, Beyond co-occurrence: discovering and visualizing tag relationships from geo-spatial and temporal similarities, in: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, ACM, 2012, pp. 33–42.