A 'glocal' approach for real-time emergency event detection in Twitter

A 'glocal' approach for real-time emergency event detection in Twitter

Dario Salza
dario.salza@linksfoundation.com

Edoardo Arnaudo
edoardo.arnaudo@polito.it
edoardo.arnaudo@linksfoundation.com

Giacomo Blanco
giacomo.blanco@linksfoundation.com

Claudio Rossi
claudio.rossi@linksfoundation.com

ABSTRACT

Social media like Twitter offer not only an unprecedented amount of user-generated content covering developing emergencies but also act as a collector of news produced by heterogeneous sources, including big and small media companies as well as public authorities. However, this volume, velocity, and variety of data constitute the main value and, at the same time, the key challenge to implement and automatic detection and tracking of independent emergency events from the real-time stream of tweets. Leveraging online clustering and considering both textual and geographical features, we propose, implement, and evaluate an algorithm to automatically detect emergency events applying a 'glocal' approach, i.e., offering a global coverage while detecting events at local (municipality level) scale.

Keywords

Emergency, Event Detection, Social Media, Twitter, Incremental Clustering

INTRODUCTION

Under emissions in line with current pledges under the Paris Agreement, global warming is expected to surpass 1.5°C above pre-industrial levels, even if these pledges are supplemented with very challenging increases in the scale and ambition of mitigation after 2030 (Allen et al. 2018). Despite this slight increase, the consequences of global warming are already observable to this day, with the number and intensity of certain natural hazards in continuous growth (e.g. extreme weather events, floods, wildfires). Social networks have often assumed a crucial role as means of mass communication during such emergencies in the recent years, thanks to their pervasive reach, counting 3.6 billion users worldwide as of 2020 (Statista 2021). They constitute a powerful and versatile bidirectional channel, allowing on one side authorities and news agencies to quickly alert and instruct a wide audience in the population; on the other, citizens to submit reports and multimedia from the field that are of inestimable value for modern journalism. These may also be extremely beneficial to first responders, decision makers and practitioners who could exploit this exhaustive and widespread data to better assess the situation and guide the decision making process. Despite social media use in emergency scenarios has been proposed for many applications in many previous studies (Imran et al. 2015), the provision of a coherent and comprehensive overview of emergency events remains a challenging task due to several reasons: first, social media represent inherently heterogeneous and noisy data, thus posing significant issues for the retrieval and collection of informative content in such critical scenarios. Second, information is seldom complete and provided in small fragments, which require careful considerations for an accurate aggregation. Last, the high volume of multimedia resources, coupled with their impressive publication rate, requires the implementation of an efficient processing pipeline, able to deal with large quantities of near real-time data. Conveniently, it is often the case that the very same social media platforms (e.g., Twitter) provide means to directly retrieve information as continuous pre-filtered streams, which facilitates the adoption of emergency management services based on such kind of data.

To address the aforementioned challenges, in this work we propose, implement, and evaluate a comprehensive methodology to automatically detect emergency events and collect relative content applying a *glocal*, holistic

approach: while on one side we draw from the worldwide, unrestricted data stream, providing a global scale coverage, on the other side we process elements at a local (municipality level) resolution, tracking smaller and fragmented events. Our system thus satisfies the following requirements:

- The system should be able to operate in real time but produce a meaningful result even in retrospective, with an unlimited number of simultaneous events. The set of tweets referencing an event should be part of the produced output, and updated as new information becomes available.
- The system should detect both large- and small-scale events: while larger disasters offer often a continuous arrival of information, tweets relative to smaller incidents usually are much more temporally sparse.
- Events should be defined by their location, type and time, and, when applicable, distinct events of the same type in the same or neighboring locations should be detected separately. Event duration should be unbounded.
- The events should be able to span over multiple locations, whether they are a large-scale disaster or a minor incident that is being reported in logically linked but different places (such as the actual location and its overarching area, e.g. county or province). The same tweet may reference more than one separate event.

Our algorithm allows to filter social media posts, removing noise and unrelated content, and to group them within candidate events, thus providing a content management technique that could make the exploitation of social media sources during ongoing crisis viable also with limited human resources. Furthermore, for certain hazards, the social media event detection could automatically trigger other downstream services aimed to perform an additional validation step. For example, the detection of a flood event could trigger an automatic water mapping algorithm using satellite Synthetic Aperture Radar (SAR) data, while the detection of a wildfire could be validated looking at remote sensed hot spots delivered from satellites such as VIIRS and MODIS. Moreover, a curated list of emergencies and incidents could support many retrospective applications, including trend and risk analysis, and simulation of historical events considering additional prevention measures.

The remainder of this paper is organized as follows. After a review of related works and precedent approaches, we describe our methodology, based on online clustering and considering both textual and geographical features. We then present the results obtained processing real world data over an 8 month period and multiple languages (English, Italian, Spanish), evaluating the outcome over multiple quality metrics and discussing future improvements.

RELATED WORK

Event detection algorithms applied to social media and especially Twitter content have been extensively studied and classified according to specific taxonomies (Atefeh and Khreich 2015, Saeed et al. 2019). A first division occurs between the detection of *unspecified* and *specified events*. In the first case, no prior information or filtering is provided, often defining an event as a topic attracting the attention of a large quantity of users, causing a surge in volume for some theme or entity in the general discussion. In the second case, several attributes of the event of interest (current, past or planned) are known beforehand, such as location, time, or involved entities. Disaster-related event detection systems often operate between these two extremes, analyzing streams filtered to include only a set of topics or cover a specific geographical area of interest, but without any information on whether, when or where an event of some kind will occur, attributes that are often the target of the detection.

A second division is the one between document-pivot and feature-pivot techniques. In the former, the full documents (i.e., the tweets) are clustered according to their content similarity, while in the latter the volumetric evolution of some unique features (single keywords or entities, locations) is analyzed and correlated. Lastly, event detection systems may operate in near real-time, attempting a New Event Detection (NED) as additional information is received from the streams, or a Retrospective Event Detection (RED), having available all the documents produced in the time frame of interest.

Feature-pivot methods

Related to the problem of unspecified NED, Fedoryszak et al. 2019 and Long et al. 2011 extract cyclically relevant and bursting entities, hashtags or keywords and place them as nodes in a clique graph where the edge weight is a measure of co-appearance of the two entities. The correlated entities are then clustered using respectively Louvain community detection and hierarchical divisive clustering, and the resulting groups are linked via Bipartite Graph Matching with those produced at the previous cycle, creating cluster chains. With a similar approach, Weng and Lee 2011 adopt wavelet transform to build entropy-based signals from the cyclical sampling of each

word *DF-IDF*, filtering away trivial ones according to their auto-correlation. The graph built from the remaining signals inter-correlation is then recursively partitioned to maximize modularity. These methods perform well in detecting and linking generic trending entities and topics, but are unable to catch minor, temporally sparse events, or distinguish similar but independent events.

Filtered-stream methods often exploit pure volumetric analysis: Sakaki et al. 2010 were able to detect earthquakes in Japan before the official announcement of the relevant agency by evaluating the number of tweets containing selected keywords and classified as event-relevant by an SVM-based model over time, using a probabilistic model to confirm spikes in tweet count. In addition, the epicenter is estimated from the available geolocated tweets using Kalman and particle filters. Leveraging only geolocated tweets, Dittrich and Lucas 2014 partition the area of interest into a grid and for every cell a time-series anomaly detection is performed on the number of tweets localized inside of it, hypothesizing a normal distribution. Considering then the tweets contained in each anomalous cell, the most represented words are extracted and matched to a multilingual list of disaster-related keywords, and same-event-type neighboring cells are merged in a unique event. Similarly, Thapen et al. 2016 detect geographically partitioned time-series volumetric anomalies, but the cells have an irregular shape and are defined with a previous clustering step, while the event topics are separated into independent streams beforehand. Considering independent tweet streams by location, Hossny and Mitchell 2018 extract the word pairs most associated with protests as a first model building step, then their daily count is used as input features to several classifiers to predict whether an event has occurred in the location in that day. Volumetric-based methods are effective and reliable for major events, but less so for minor or temporally sparse ones, and may be incapable or require significant resources to distinguish neighboring same-type independent incidents.

Document-pivot methods

Considering only textual information, Hasan et al. 2016 apply a two-stage process looking for newsworthy events: in the first step an incoming tweet is compared against a buffer of recent ones, forwarding it to the second step only if it has at least one match with cosine similarity over a threshold. In the second stage, tweets are clustered in incremental and greedy fashion, attaching a tweet to a cluster if its cosine similarity with the centroid is again over a threshold, and merging clusters that match strongly with the same single tweet. Each stage has a lookup map between the most informative keywords (determined via an incremental TD-IDF) and the tweet or cluster match candidates, limiting the number of comparisons. A cluster is upgraded to event if contains at least 10 items and satisfies additional criteria such as containing a link to a news portal. Petrović et al. 2010 and McCreadie et al. 2013 follow a similar incremental clustering approach, leveraging Locality Sensitive Hashing to evaluate the novelty and group in the same bucket similar tweets. Liu et al. 2016 use a stack of filters based on fuzzy rules, topic modeling and classification algorithms to isolate tweets regarding breaking news, and perform a first aggregation to create unit clusters of 3 tweets that contain similar Named Entities, hashtags and textual features. The unit clusters are then further filtered by topic-level credibility and merged to compatible clusters contained in a 24 hour cache. Once generic tweet clusters are composed from unfiltered streams, the type of referenced event can be determined evaluating and matching their keywords against those recurring in disasters, as performed by Angaramo and Rossi 2018, or more advanced classifiers. Addressing specifically emergency events, Klein et al. 2013 use a Named Entity Recognition (NER) model to extract from each tweet an emergency category given predefined keywords and a location or an event name (e.g. Hurricane Sandy), then tweets with same tuple values are grouped together, comparing locations also along administrative hierarchies.

Incorporating geographical features in a RED clustering process, the approach followed by Ghaemi and Farnaghi 2019 is to apply and expand the VDBSCAN algorithm to group geolocated tweets, employing exponential spline interpolation to determine the different geographical search radiuses and imposing a minimum value of the textual similarity for geographically neighboring elements to be clustered together. In a near real-time context, Zhou and L. Chen 2014 propose LTT, an expansion of LDA taking in consideration beyond text content the location coordinates and time to represent a tweet as a probability distribution over a fixed number of topics (periodically refreshed), with the distance between messages computed as a combination of the bidirectional KL Divergence between the two distributions and the user link similarity. The clustering of similar elements is then accelerated through an optimized hash structure, considering hourly time slots, showing an application during Cyclone Ului and the Queensland floods. Lastly, Becker et al. 2010 show an application of single pass threshold-based incremental clustering to group related Flickr post using multiple and diverse features (textual content, tags, location, upload time), evaluating and discussing the performance of both ensemble and learned similarity measures. Document-pivot methodologies can detect minor events but can be complex to scale and suffer from the high quantity of noise and ambiguity present in social media content. Taking into consideration geospatial features helps to disambiguate similar but independent events and to partition or index intuitively the items, leading to improved scaling.

SYSTEM DESCRIPTION

The overall pipeline is based on two main concepts: *clusters* and *events*. The former simply represents a collection of similar documents and it is exploited as internal data structure, the latter can be seen as a promoted (*flagged*) cluster that satisfies a set of required criteria, and constitute the output of the system. The general schematic of the proposed system, parallel and independent for each language and hazard type combination, is reported in Figure 1, and follows in the first part a greedy incremental clustering approach: tweets, received from a streaming source or in batches, are individually preprocessed (**Tweet Preprocessing**) and attached to one or multiple clusters (**Incremental Clustering**). Cyclically, this process is suspended, and several cluster management operations are performed, such as **Cluster Defragmentation**, that finds and merges independent active clusters that may describe the same event, having over time become similar or neighboring. In these first two steps all clusters, whether flagged as event or not, have no difference in behavior. Subsequently, non-event clusters are evaluated for activation (**Event Activation**), after which their content is purged if expired and they are removed if remained empty (**Tweet and cluster expiration**).

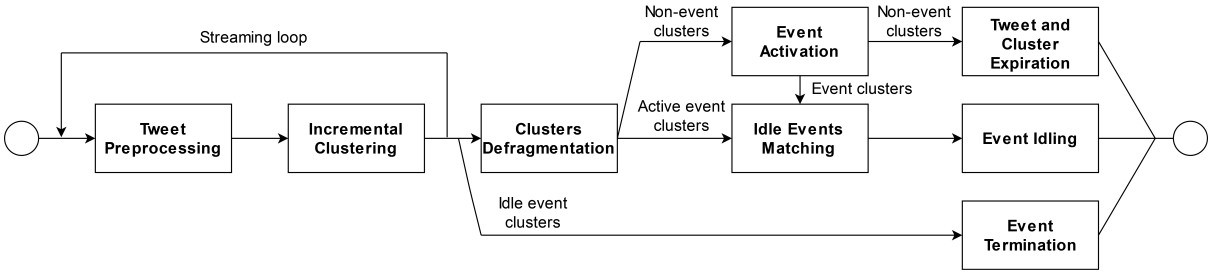

Figure 1. Event detection pipeline, highlighting the major processing modules.

Regarding events, the system keeps in memory two sets: *active* and *idle* events. While the first are full clusters that continue to accumulate tweets in the incremental clustering and defragmentation phases, the second ones are excluded from these processes and keep a reduced representation in memory. Active events are converted into idle ones when they reach a certain age (**Event Idling**), and these last are similarly removed from memory after a timeout (**Event Termination**). Active events are periodically evaluated against idle ones, and, when matching, they inherit their identifier, thus being set as a continuation of the same event (**Idle Events Matching**). After the completion of all cluster management tasks the system loops, resuming the tweet preprocessing and clustering operations.

Tweet Preprocessing

A data ingestion module interfaced with Twitter's streaming API (Twitter, Inc. 2021) continuously receives messages in real time, querying for a list of emergency and disaster-related keywords for each language-hazard pair. The retrieved tweets are immediately associated to one or more hazard types, including but not limited to those in Table 1, according to the same keywords and thus allowing to partition the following steps by topic. Their textual content is then cleaned from emojis, urls and unwanted characters, and tokenized. Each token is then mapped to the equivalent pretrained MUSE multilingual embedding (Conneau et al. 2017), and the Convolutional Neural Network model described in Piscitelli et al. 2021 classifies the tweet text's information type, allowing to discard in the following steps every element evaluated as *Irrelevant Information* to both remove noise and reduce volume. Subsequently, a NER model based on the LSTM Neural Network described in Lample et al. 2016 and trained on the Ontonotes dataset (Weischedel et al. 2012) using the same MUSE embeddings extracts the location names, that are geocoded to the relative longitude and latitude bounding box using Nominatim (OpenStreetMap contributors 2021), and date entities, that are parsed to the corresponding timestamp. A bounding box is considered rather than its centroid point since the latter does not allow to compute distance correctly between neighboring or overlapping wide areas. Tweets with no geographical information are ignored.

The tokenized textual content is then lemmatized and POS-tagged using the spaCy library (Honnibal et al. 2020), removing non-keyword tokens using a set of accepted POS tags rather than relying on a fixed list of stop words to remove (Table 1). In addition to this, we also remove some words that, while part of the allowed POS, are not meaningful keywords useful to infer a distinction between different events (such as 'breaking news'). In case of foreign or proper nouns, a different lemmatization is sometimes extracted for the same word, therefore the original token is kept as an alias along with the computed lemma. A single tweet is therefore represented as its tweet and author id, information and hazard type, a set of unique keyword lemmas and hashtags, a list of geographical bounding boxes and timestamps.

Detected Hazard Types	Storm; Landslide; Avalanche; Terrorism; Extreme temperatures; Earthquake; Flood; Wildfire
Information Types	Affected People; Caution and advice; Infrastructures and utilities; Donations, volunteering or rescue efforts; Other useful information; Irrelevant information
Allowed POS Tags	ADJ - Adjective; NOUN - Noun; NUM - Numeral; PROPN – Proper Noun; VERB - Verb; X - Other

Table 1. Taxonomies for detected Hazard Types and Information Types, and allowed Part of Speech (POS) Tags.

Centroid structure and similarity metrics

Cluster centroids aggregate the content of the belonging tweets into three separate bag of words (that is, dictionaries containing entity and number of references respectively as key and value), one for each of the features considered: textual keywords, hashtags and locations. Additionally, they also keep the set of the unique ids of the documents composing the cluster. When comparing a document or a cluster to another cluster, a similarity score for each feature is computed independently, and then composed with a standard weighted sum:

$$sim = \sum_{f \in \{t,h,g,c\}} \lambda_f \cdot f$$

where t, h, g and c respectively represent textual, hashtag, geographical and *common element* score. A constraint on the minimum acceptable value is set for the text score, θ_{ts}, otherwise the similarity score is set to zero regardless of the actual value of the sum. This allows for geographically neighbouring hazards to have a lower textual similarity, given that the probability of two events of the same type happening simultaneously is low, but requires a higher score for larger scale distances or areas. All parameters values can be defined separately for each hazard type or language, and although most values are otherwise shared across instances, the parameters most tied to the geographic features and the plausible event scale indeed change between wide (e.g. earthquakes) and narrow (e.g. avalanches) disaster types.

With text and hashtag BoW converted into a vector format, indexed by entity and valued by number of references, the text score and hashtag score of two compared elements c_A and c_B are computed as the simple cosine similarity between each pair of vectors. While the first element is always a centroid, the second can be both a centroid or a document, in which case the number of references for each of its entities is considered to be equal to 1. Concerning the g score, defining a distance between polygons is a non-trivial task compared to the point-based case, as the metric must convey not only a measurement of neighborhood, but also of same-event likelihood, which intuitively should be smaller as the area of interest becomes wider. Additionally, a cluster centroid can contain more than one location, therefore given a centroid c_A and its most referenced location $l_{c_A}^{max}$, only the set of locations belonging to c_A whose number of references is over the absolute threshold $T = \theta_{cl} \cdot ref(l_{c_A}^{max})$ is considered. Then, given all the possible permutation pairs of valid locations $l_{c_A}^i \in c_A \times\ l_{c_B}^j \in c_B$, the value of each pairwise $g_{i,j}$ score is computed as:

$$d_{i,j} = \phi_{ds}\left(D_{min}(l_{c_A}^i, l_{c_B}^j)\right)$$

$$L_{i,j} = \phi_{ls}\left(\sqrt{\max\left(A(l_{c_A}^i), A(l_{c_B}^j)\right)}\right)$$

$$g_{i,j} = \min(d_{i,j}, L_{i,j})$$

excluding the transformation ϕ, $d_{i,j}$ computes the minimum approximated Haversine distance between $l_{c_A}^i$ and $l_{c_B}^j$, and is thus zero if the two locations are the same or overlap; $L_{i,j}$ represents the likelihood score, dependent on the area of the largest of the two polygons; finally the ϕ_t function inverts and non-linearly shrinks (i.e., a greater surface or distance produce a smaller score) the raw measurements to the $[0, i_g]$ range, where the upper threshold $i_g \leq 1$. The transformation is defined as:

$$\phi_t\,(x) = \min(i_g, e^{-\gamma_{1,t}\cdot(x+\gamma_{2,t})^2})$$

At this stage a constraint on the minimum distance is also enforced, and if $D_{min}(l_{c_A}^i, l_{c_B}^j)$ is not within the range $[0, \theta_{md}]$, $g_{i,j}$ is set to zero. Last, the final geographical score g is set to the maximum value between all the pairwise scores $g_{i,j}$.

Finally, the common element score c is only defined when comparing generic clusters, A and B, with at least three tweets each, and refers instead to the ratio of common items between the two:

$$c = \max\left(\frac{|A \cap B|}{|A|}, \frac{|A \cap B|}{|B|}\right)^2.$$

Whenever c is greater than a predefined threshold, θ_{ces}, the clusters are considered to be correlated independently of their geographical distance, which may also be the result of a geocoding error, and the constraint on the value of the minimum distance is ignored.

Incremental Clustering, Defragmentation, and Idle Event Matching

The clustering process is parallel and independent for every language and hazard type, and it follows the schematic of two-pass incremental clustering: an element is first assigned to a cluster with highest similarity score that is over a predefined threshold θ_{tc}, otherwise a new cluster is created. Then, to reduce fragmentation, recently modified clusters are periodically compared and merged if their similarity is over a second threshold θ_{cca}.

Since a single tweet may be about more than one independent event and thus point to more than one location, the first assignment is not performed directly on tweets, but rather once per named location: each surrogate document used in the comparison, named *hint*, has the same identifier, text content and hashtags of the source post, but only one of its geographic entities (see algorithm 1). The weight of each hint is equal to

$$w = \frac{1}{count(locations \in tweet)}$$

and is exploited when updating the centroid after the assignment: for each feature's BoW, the number of references for all entities contained in the hint is increased by the weight w. In practice, if all the hints created from a tweet are assigned to the same cluster, the references of the relevant entities all increase exactly by 1, except for the geographical ones, that are not repeated between hints. However, it is also possible that hints from the same tweet are assigned to separate clusters, thus the number of entity references in the centroids can be fractional. For both hint assignation and cluster defragmentation, the candidates for comparison are selected and filtered by geographical proximity using an index, reducing the number of operations and improving performance.

Algorithm 1 Incremental clustering

```
clusters = []
for tweet in stream(hazard, language) do
  for location in tweet.locations do
    h = hint(tweet.id, tweet.keywords, tweet.hashtags, location)
    h.weight = 1 / len(tweet.locations)
    candidates = get_neighbours(clusters, location)
    best_cluster = argmax(similarity(h, c) for c in candidates)
    best_score = similarity(hint, best_cluster)
    if best_score < θ_tc then
      best_cluster = cluster()
    end if
    recompute_centroid(best_cluster, h)
    update(clusters, best_cluster)
  end for
end for
```

While all the clusters recently modified, flagged as event or not, are considered in the defragmentation process, only those that have been activated are evaluated in the Idle Events Matching step. In this additional defragmentation operation, whenever the similarity between a currently active event and an idle one is higher than θ_{cci}, the active cluster inherits the identifier and starting time of the older one (and as such is marked as its continuation), while the older one is removed from memory, preventing further linking to the same event.

Event Activation

All clusters modified during the current cycle and not flagged as event yet are evaluated and activated if all the following constraints are satisfied:

- The number of unique tweet authors in the cluster is higher than a small threshold (3 for Italian, 5 for Spanish and English), given that this metric was found in Petrović et al. 2010 to be more reliable than the simple number of tweets to assess the relevance of a topic.
- Given the hash of the text content of each tweet, there is a collision (i.e., two tweets share the same text) for less than 25% of the tweets in the cluster, to counteract uninformative clusters (such those formed by calls to action, petitions and generic copy-and-paste content).
- Less than 70% percent of the tweets belong to the information type categories *Caution and advice* or *Donation and volunteering efforts*, to prevent activation of clusters relative to an event yet to happen or that contain little information on the event itself.
- The number of date entities referencing earlier than the previous week or in the future are less than those referencing the current date, to prevent the activation of clusters referencing past events.

A cluster can be evaluated for activation in an unlimited number of cycles, as long as it gets updated and modified by the inclusion of new tweets or the expiration of older ones.

Tweets and Cluster Expiration, Event Idling and Termination

Clusters must be activated and flagged as event within a time τ_{act} from the timestamp of the earlier tweets in the cluster, otherwise all the relative hints are removed from the cluster, and their entities references are subtracted from the centroid. If, after this removal, the cluster remains without tweets, it is then completely removed from memory.

Similarly, upon activation, the timestamp of the first tweet is set as the event start time, and the cluster lasts only a fixed duration time τ_{dur}. After this time has elapsed, the event is subsequently removed from the list of active clusters and added to the idle ones. Since the continuation of the event is dependent on Idle Events Matching step and thus on the existence of a second and compatible cluster, to help the aggregation to a unique cluster of new incoming tweets related to the same event, a spinoff cluster is created. This cluster is equal to new empty clusters and is not linked in any way to the originating one, except that is created with an already instantiated centroid, a reduced copy of the original one produced by the process shown in algorithm 2, instead of an empty one.

Algorithm 2 Spinoff Instantiation

```
Require: original
  spinoff = cluster()
  oc = original.centroid
  sc = spinoff.centroid
  M = max(keyword.references for keyword in oc.keywords)
  for keyword in oc.keywords do
    keyword.references = (σ_tx / M) · keyword.references
    if keyword.references > θ_sp then
      sc.keywords.add(keyword)
    end if
  end for
  sc.hashtags = oc.hashtags.sort_descending(by h.references for h).take(σ_hg)
  sc.geos = oc.geos.sort_descending(by g.references for g).take(σ_hg)
  sc.tweet_ids = empty_set()
  return spinoff
```

This idling with spinoff-matching and continuation allows the events to have an unbounded lifespan and to change over time, while decreasing the probability of a cluster slowly becoming boundless and continuous in time, an eventuality observed in Angaramo and Rossi 2018. Finally, after a total $\tau_{tot} = \tau_{dur} + \tau_{end}$ from its start time, if the event is still present in the idle event list, it is terminated and removed from memory, setting as end mark the time of the last tweet associated with it.

EXPERIMENTS AND EVALUATION

Case study

As a case study for the proposed system, we fetched and clustered in real time tweets containing a set of predefined keywords from January 1 2021 to August 31 2021, considering tweets in the English, Spanish and Italian languages.

Tweet id	Time	Text
1361100018439094276	14/02/2021 23:48	snowboarder touring solo Sun am on Pat's Knob / Mt Trelease north of Loveland Pass buried / killed in an avalanche. Rescuers spotted his deployed airbag in debris. And snowmobiler killed in slide near Rollins Pass. 10 ppl killed in slides this grim season. https://t.co/mzTz24gRT8
1361106329151053827	15/02/2021 00:13	Sadly @COAvalancheInfo is reporting 2 #avalanche fatalities in the Front Range Zone on Sunday, Feb 14. A snowmobiler was killed on Mt Epworth, W. of Rollins Pass. A backcountry snowboarder was killed on Mt Trelease, N. of Loveland Pass. https://t.co/G6PxgjHrN2 #COWX
1361121859736109059	15/02/2021 01:15	Backcountry Snowboarder Caught & #Killed In Avalanche Near Loveland Pass - Feb 14 @ 8:13 PM ET https://t.co/2Rjp0U68AT
1361140910533263365	15/02/2021 02:30	Backcountry Snowboarder Caught & Killed In Avalanche Near Loveland Pass – CBS Denver - https://t.co/EeQ5AONw68 #RockyDailyNews https://t.co/X3WUrcuBeo
1361293211168346113	15/02/2021 12:36	Backcountry Rider Caught & Killed In Avalanche Near Loveland Pass https://t.co/a308ajLiQ9 via @YouTube

Table 2. Detail of a specific avalanche event from February 2021. Despite the relatively small amount of tweets, the automated detection correctly identified the news and clustered them together by similarity and location.

Each country was covered by exactly one language: in Spanish and Italian speaking countries only the events created from tweets in these languages were saved, while English covered the rest of the world.

To tune the various hyperparameters, we used tweets collected between November and December 2020 and manually grouped to define the distribution of values of the similarity metrics among same-event and different-event tweets to define a first rough estimate, then we tweaked empirically the values with multiple runs of the system over the same period.

In the 8-month period of evaluation, 28 million tweets were retrieved in total, and after the preprocessing phase (filtering by informativity and presence of a location) 8.5 million were served as input to the clustering steps and following, creating a total of 32 thousand events, composed by more than 2 million tweets. Of the total amount of tweets, only 0.75% was geotagged, an even lower figure than the 2% reported previously by Leetaru et al. 2013. The chosen cycle time between cluster management operations was of 5 minutes, and the peak tweet count per cycle was 5331 tweets the 30/07 in the morning, when the Turkish wildfires gained international attention.

While the 49% of events was composed by less than 10 tweets, 14 events contained more than 10 thousand, concerning the Haiti Earthquake of the 14 August 2021, the Turkish wildfires between the end of July and the beginning of August, several episodes of terrorism and arson in Myanmar, Hurricane Ida between August and September, and finally the 6 January protests in Washington DC, which were incorrectly flagged as storm due the frequent use of the sentence 'storm the Capitol'. On the opposite, minor events (such as the one reported in Table 2) were for example relative to smaller and localized incidents, or weaker earthquakes in remote areas.

Finally, 88% of the events had a duration of less than 1 day, while the longest events were what was flagged as a 25-day cold wave in the UK (January 2021 was indeed the coldest month in the area since 2010) and again the 6 January protests, which attracted an uninterrupted discussion for 43 days.

Performance

The experiments were run on a virtual machine with 32GB of RAM and a Intel Core i9-7940X CPU, with the full pipeline composed by five modules developed in Python running on separate processes and interconnected through RabbitMQ or Python IPC queues:

- Tweet intake and tokenization, with negligible latency and high throughput
- Information classification, with a throughput of 4500 tweets/second
- Named Entity Recognition, entity linking via Nominatim and date parsing, with a throughput of 200 tweets/second

- Final preprocessing (including spaCy NLP) operations, with a throughput of 120 tweets/second
- Incremental clustering, performed in batches, and cluster management operations, with a total maximum latency (input of a tweet in the module to potential event detection) equal to the cycle or batching interval (5 minutes) plus the effective processing time per batch. This latter, aggregated by sum for all hazard-language pairs, was on average 3.2 seconds and had a peak of 10.1 seconds, depending on the number of tweets to process and of clusters in the system. It would be possible to remove the batching interval from the latency by operating in streaming mode and executing the event activation task after every cluster modification, instead of once per cycle. This module had a maximum aggregated RAM consumption of 1GB.

While for the input volumes considered in our case study the throughput of all modules was highly sufficient even during inflow peaks, there may be some bottlenecks when operating with 10x to 100x elements. In this case, two main actions could improve the system scalability. First, it must be noted that while all the NLP models (such as spaCy and the NER) in our study were executed on CPU, deploying them on GPU could offer a considerable throughput improvement. Second, currently all modules cover with a single instance and a single worker thread all language-hazard pairs, and only the data structures are replicated independently. A separate full pipeline instance for each targeted combination should increase the full throughput by an order of magnitude, and even more by simply replicating all the stateless modules that may be a bottleneck.

Experiment setup

To validate the quality of the system output, we first sampled randomly 25 events (or as many available if less), and all the relative tweets, for each language and hazard type, for a total of 456 events and 76876 tweets. We then evaluated manually each cluster over multiple criteria:

- Content type: whether the cluster is regarding a breaking disaster-type event and its developments (Event); is generic discussion regarding a recent event or some aspects of it (Event-Relative); concerns an event of the past, such in the case of an anniversary or breaking news due a new development, such as a trial ruling (Past Event); is relative to a forecast, warning or precautions against an expected or probable future event (Event Alert); is not a disaster-type event, but an unrelated breaking news, discussed topic or generic noise (Not event).
- Coherency: independently from the content, if all the tweets in the cluster appear to be on the same topic (Coherent); if most of them are, and it is possible to understand a main discussion topic, but there are some evident outliers (Coherent with noise); if the cluster appears to be coherent at first, but closer inspection reveals multiple sub-topics, such as close subsequent shakes in an earthquake event or multiple neighbouring landslides or floodings during the same intense storm (Coherent with overlapping subevents); if the cluster topic is unclear, or contains multiple evidently independent events or topics (Not coherent).
- Hazard/disaster type: discarding all not-event clusters, whether the disaster type of the cluster is tagged correctly (Correct type); is not correct but is one of the other considered disaster types (Wrong type); is not correct and relative to a disaster type outside of the considered ones, such as a volcanic eruption, an explosion, or a plane crash (Other events).
- Location correctness: discarding all not-event clusters and considering as event location the most referenced location entity in the cluster, and the corresponding bounding box as extracted by the pre-processing pipeline, it was evaluated if it was the correct one (Correct location); a location near or containing the correct one (Imprecise location); if the location was incorrect due an error of the NER (Incorrect named entity extraction); if the location name is correct, but the geocoding produced an homonym location in a different region, or even a completely unrelated result (Incorrect geocoding).

Additionally, to verify the ability of the system to detect events, we extracted from the EM-DAT disaster database (EM-DAT 2009) the list of incidents in the considered period, removing from the list those with an imprecise start date or a disaster type not among the considered ones. Afterwards, we used Nominatim to geocode all the bounding boxes corresponding to the entities named in the 'Location' column, merging them in one target multi-polygon area for each event. We then computed how many of the EM-DAT events matched to one or more activated clusters produced by our system, considering clusters with the same disaster type, a location intersecting the target one, and a starting date between the start date and up to 48 hours after the end date reported by EM-DAT. We then repeated the process for the disaster types reported in the associated disaster column.

A. Content type	Absolute	Relative
Event	239	52%
Event-relative	83	18%
Past event	11	2%
Event alert	52	11%
Not event	71	16%
TOT	456	

B. Coherency	Absolute	Relative
Coherent	397	87%
Coherent with noise	39	9%
Coherent with overlapping subevents	10	2%
Not coherent	10	2%
TOT	456	

C. Hazard type	Absolute	Relative
Correct type	365	95%
Wrong type	11	3%
Other events	9	2%
TOT	385	

D. Location correctness	Absolute	Relative
Correct location	308	80%
Imprecise location	30	8%
Incorrect named entity extraction	12	3%
Incorrect geocoding	35	9%
TOT	385	

Table 3. Quality metrics in respect to Content Type (A), Coherency (B), Hazard Type (C) and Location Correctness (D).

Subset	Detected	Total	Percentage
Global disasters	180	259	69%
Disasters in English, Spanish, Italian speaking countries	55	77	71%
Global associated disasters	53	97	55%
Associated disasters in English, Spanish, Italian speaking countries	17	35	49%

Table 4. Detection rates computed over the EM-DAT dataset.

Result analysis and future works

While many aspects of the performance of the system can or should be ascribed to the filtering, NER and geocoding upstream pipeline, we find that the metric most relevant to define the approach quality, the cluster coherency, has a satisfiable value, having 96% of the clusters with little to no noise. This confirms that the incremental clustering algorithm and the scoring technique used in this work perform well in the task of grouping correlated documents. On the other hand, a ratio of 52% of real events over the total amount of events produced means that too many false positives or false alarms are part of the output: while an overhaul of the upstream tweet informativeness classifier could help, its maximum performance is bounded by the ambiguous and uncontextualized nature of the tweets. As such, an additional downstream NLP filter operating on the entire clusters of tweets may help in removing false positives, especially those of the categories *Past event*, *Event alert* and *Not event*.

Regarding the around 30% of EM-DAT registered disasters that were not detected, we found it most of the time to be the result of two factors: firstly, the imperfect performance of the NER and geocoding preprocessing steps, which combined may place a correctly detected event in an imprecise or incorrect location once every five times, and must be one of the focuses for future improvement; secondly, events appearing in the EM-DAT can have a slow evolution or occur in remote areas, resulting in a minimal, delayed or extremely temporally sparse online participation, which in turn hinders or impedes their detection. On the other hand, it must be noted that estimating conservatively one every two produced events to be a real one, our system detected more than 15000 incidents and disasters over the entire globe, almost 60 times the number of those registered by EM-DAT, given its ability to detect and archive local and minor events that are omitted by global disaster databases.

Finally, while we found the parameter tuning phase to be feasible using a mixed statistical-empirical process, and that small changes did not disrupt the performance, given the high number of independent parameters and their nature is hardly possible to guarantee that the tuning found is globally optimal. Several strategies could help in this regard, such as exploiting a black box machine learning model as a similarity scoring function, which could replace the current model without requiring any change to the clustering algorithms. However, its training requires a labeled dataset with the correct mix of local and global, small and large events and noise, whose creation is non-trivial and expensive.

CONCLUSIONS

We discussed in this paper an approach to automatically detect in real time incidents and emergencies using social media, specifically Twitter, content. We presented an overview of the preprocessing pipeline and proposed an

incremental clustering algorithm, a similarity metric and an activation and termination strategy suitable for this challenge, targeting a local resolution in a global scale, allowing for an unbounded number of events of unbounded duration, striving to detect small or temporally sparse incidents. The similarity scoring function exploits not only textual similarity but also geographical proximity, handling multiple bi-dimensional regions and locations rather than assuming a rarely available point geotag. We then evaluated our approach over an 8-month period and multiple languages, extracting tens of thousand of clusters of interest, and assessed various quality metrics of the result, highlighting the current strengths and shortcomings of our methodology, delineating consequently possible solutions and future work.

Acknowledgments *This work was developed in the context of the Horizon 2020 projects FASTER (grant agreement n.833507), SAFERS (grant agreement n.869353) and APPRAISE (grant agreement n.101021981).*

REFERENCES

Allen, M. R., Babiker, M., Chen, Y., Coninck, H. de, Connors, S., Diemen, R. van, Dube, O. P., Ebi, K. L., Engelbrecht, F., Ferrat, M., et al. (2018). *Global Warming of 1.5: An IPCC Special Report on the impacts of global warming of 1.5C above pre-industrial levels and related global greenhouse gas emission pathways, in the context of strengthening the global response to the threat of climate change, sustainable development, and efforts to eradicate poverty. Summary for policymakers*. Tech. rep. IPCC.

Angaramo, F. and Rossi, C. (2018). "Online clustering and classification for real-time event detection in Twitter." In: *ISCRAM*.

Atefeh, F. and Khreich, W. (2015). "A survey of techniques for event detection in twitter". In: *Computational Intelligence* 31.1, pp. 132–164.

Becker, H., Naaman, M., and Gravano, L. (2010). "Learning similarity metrics for event identification in social media". In: *Proceedings of the third ACM international conference on Web search and data mining*, pp. 291–300.

Conneau, A., Lample, G., Ranzato, M., Denoyer, L., and Jégou, H. (2017). "Word Translation Without Parallel Data". In: *arXiv preprint arXiv:1710.04087*.

EM-DAT (2009). *EM-DAT: The International Disaster Database*. http://www.emdat.be/.

Dittrich, A. and Lucas, C. (2014). "Is this Twitter Event a Disaster?" In: *17th AGILE Conference on Geographic Information Science*.

Fedoryszak, M., Frederick, B., Rajaram, V., and Zhong, C. (2019). "Real-time event detection on social data streams". In: *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2774–2782.

Ghaemi, Z. and Farnaghi, M. (2019). "A varied density-based clustering approach for event detection from heterogeneous twitter data". In: *ISPRS international journal of geo-information* 8.2, p. 82.

Hasan, M., Orgun, M. A., and Schwitter, R. (2016). "TwitterNews+: a framework for real time event detection from the Twitter data stream". In: *International conference on social informatics*. Springer, pp. 224–239.

Honnibal, M., Montani, I., Van Landeghem, S., and Boyd, A. (2020). *spaCy: Industrial-strength Natural Language Processing in Python*.

Hossny, A. H. and Mitchell, L. (2018). "Event detection in twitter: A keyword volume approach". In: *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, pp. 1200–1208.

Imran, M., Castillo, C., Diaz, F., and Vieweg, S. (June 2015). "Processing Social Media Messages in Mass Emergency: A Survey". In: *ACM Comput. Surv.* 47.4.

Klein, B., Castanedo, F., Elejalde, I., Lopez-de-Ipina, D., and Nespral, A. P. (2013). "Emergency event detection in twitter streams based on natural language processing". In: *Ubiquitous Computing and Ambient Intelligence. Context-Awareness and Context-Driven Interaction*. Springer, pp. 239–246.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). "Neural Architectures for Named Entity Recognition". In: *CoRR* abs/1603.01360. arXiv: 1603.01360.

Leetaru, K., Wang, S., Cao, G., Padmanabhan, A., and Shook, E. (2013). "Mapping the global Twitter heartbeat: The geography of Twitter". In: *First Monday*.

Liu, X., Li, Q., Nourbakhsh, A., Fang, R., Thomas, M., Anderson, K., Kociuba, R., Vedder, M., Pomerville, S., Wudali, R., et al. (2016). "Reuters tracer: A large scale system of detecting & verifying real-time news events from twitter". In: *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pp. 207–216.

Long, R., Wang, H., Chen, Y., Jin, O., and Yu, Y. (2011). "Towards effective event detection, tracking and summarization on microblog data". In: *International conference on web-age information management*. Springer, pp. 652–663.

McCreadie, R., Macdonald, C., Ounis, I., Osborne, M., and Petrovic, S. (2013). "Scalable distributed event detection for twitter". In: *2013 IEEE international conference on big data*. IEEE, pp. 543–549.

OpenStreetMap contributors (2021). *OpenStreetMap Nominatim*. https://nominatim.openstreetmap.org.

Petrović, S., Osborne, M., and Lavrenko, V. (2010). "Streaming first story detection with application to twitter". In: *Human language technologies: The 2010 annual conference of the north american chapter of the association for computational linguistics*, pp. 181–189.

Piscitelli, S., Arnaudo, E., and Rossi, C. (2021). "Multilingual Text Classification from Twitter during Emergencies". In: *2021 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE, pp. 1–6.

Saeed, Z., Abbasi, R. A., Maqbool, O., Sadaf, A., Razzak, I., Daud, A., Aljohani, N. R., and Xu, G. (2019). "What's happening around the world? a survey and framework on event detection techniques on twitter". In: *Journal of Grid Computing* 17.2, pp. 279–312.

Sakaki, T., Okazaki, M., and Matsuo, Y. (2010). "Earthquake shakes twitter users: real-time event detection by social sensors". In: *Proceedings of the 19th international conference on World wide web*, pp. 851–860.

Statista (2021). *Number of social network users worldwide from 2017 to 2025*. https://www.statista.com/statistics/278414/number-of-worldwide-social-network-users/.

Thapen, N., Simmie, D., and Hankin, C. (2016). "The early bird catches the term: combining twitter and news data for event detection and situational awareness". In: *Journal of biomedical semantics* 7.1, pp. 1–14.

Twitter, Inc. (2021). *Twitter API Reference*. https://developer.twitter.com/en/docs/api-reference-index.

Weischedel, R., Pradhan, S., Ramshaw, L., Kaufman, J., Franchini, M., El-Bachouti, M., Xue, N., Palmer, M., Hwang, J. D., Bonial, C., et al. (2012). "OntoNotes Release 5.0". In.

Weng, J. and Lee, B.-S. (2011). "Event detection in twitter". In: *Proceedings of the International AAAI Conference on Web and Social Media*. Vol. 5. 1.

Zhou, X. and Chen, L. (2014). "Event detection over twitter social media streams". In: *The VLDB journal* 23.3, pp. 381–400.

APPENDIX

Parameters

λ_t	1
λ_h	2
λ_g	0.2
λ_c	1
θ_{ts}	0.12
θ_{md} (Wide)	150km
θ_{md} (Other)	50km
θ_{ces}	0.4
θ_{cl}	0.4
i_g	0.6

$\gamma_{1,ds}$ (Wide)	15
$\gamma_{1,ds}$ (Other)	10
$\gamma_{2,ds}$ (Wide)	1.2e-4
$\gamma_{2,ds}$ (Other)	8e-4
$\gamma_{1,ls}$	190
$\gamma_{2,ls}$	1.1e-5
θ_{tc} (Wide)	0.7
θ_{tc} (Other)	0.85
θ_{cca} (Wide)	1.05
θ_{cca} (Other)	1.3

θ_{cci}	1.8
τ_{act} (Earthquake)	15 minutes
τ_{act} (Terrorism)	1 hour
τ_{act} (Other)	6 hours
τ_{dur}	24 hours
τ_{end}	18 hours
σ_{tx}	3
θ_{sp}	0.2
σ_{hg}	4

Table 5. Values for the tuned parameters adopted for the experiments shown in this paper. The 'Wide' option corresponds to the *Earthquake*, *Storm* and *Extreme temperatures* hazard types.

Hazard keywords

Avalanche	avalanche; avalanches; icefall; icefalls
Wildfire	forest fire; forest fires; wildfire; wildfires; bushfire; bushfires; conflagration; high flames; burned; explosion fire; fire firefighter; wildfire firefighter; fire firefighters
Storm	storm rain; storm rains; storm wind; storm winds; winter storm; summer storm; autumn storm; storm lightning; storm lightnings; severe storm; incoming storm; spring storm; cloud storm; storm clouds; eye storm; storms; heavy rain; heavy rains; lightnings; thunderstorm; thunderstorms; thunder storm; thunder storms; windstorm; windstorms; wind storm; wind storms; snowstorm; snow blizzard; blizzards; strong wind; hurricane; tornado; typhoon; rainfall; hurricane category
Extreme temperatures	heatwave; hot weather; hot summer; cold weather; cold winter; extreme weather; extreme temperatures; extreme cold; extreme hot; hottest summer; hottest weather; coldest winter; coldest weather; drought
Earthquake	earthquake; earthquakes; seismic; magnitude; epicentre; epicenter; building collapsed; quake victims
Landslide	landslide mud; landslide rain; landslide buried; landslide kills; landslide erosion; mudslide; mudslides; mudflow; mudflows; debris fall
Flood	flood; floods; flooding; floodings; deluge; inundation; inundated
Terrorism	terrorist attack; terrorists attack; terrorism attack; terrorist deaths; terrorist injured; terrorist hostages; terrorists dead; terrorist bomb; terrorism bomb;

Table 6. Example of hazard keywords for the English language queried to Twitter's API.

Statistics and distributions

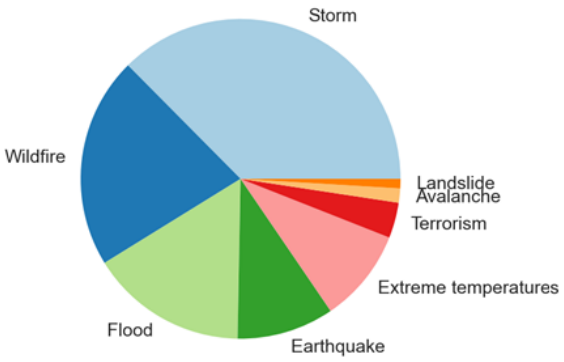

Figure 2. Overall distribution of event types, displaying a prevalence of storm, wildfires and floods over the other.

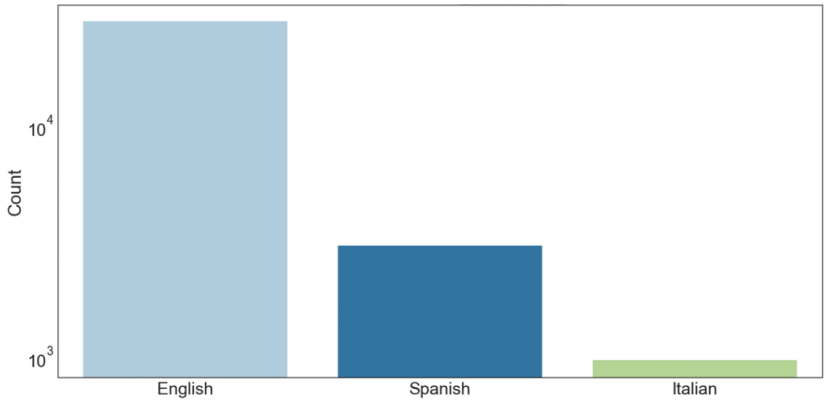

Figure 3. Overall distribution of the events per language, among the ones considered. Given the worldwide coverage, the English language is prevalent.

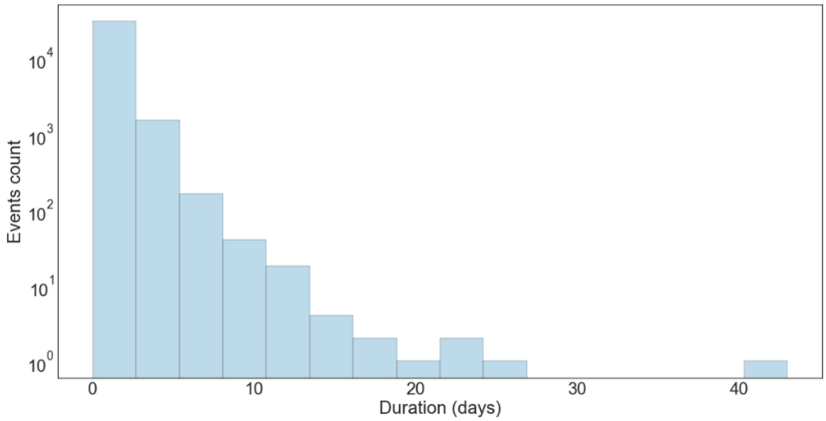

Figure 4. Overall distribution of the durations of the detected event: the trend suggests that common events do not usually go past the 20 days mark.

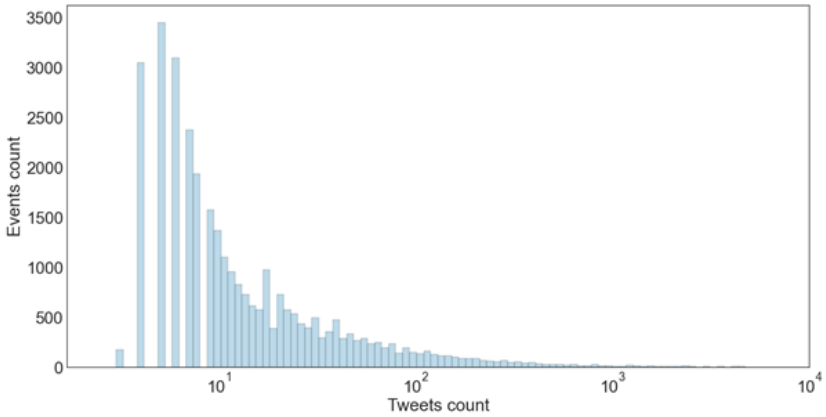

Figure 5. Overall distribution of event sizes, in terms of related tweets. As expected, the trend highlights a power law, with many smaller events and fewer, large-scale events with thousands of tweets.