15

Earthquake Observation by Social Sensors

Takeshi Sakaki and Yutaka Matsuo
The University of Tokyo
Japan

1. Introduction

Many studies have examined observation and detection of earthquakes using physical sensors. These systems require highly accurate physical sensors located over a broad area, necessitating great expense to set up the supporting infrastructure.

Social media have garnered much attention recently and the number of social media users has been increasing. Social media are kinds of media for social interaction among users. Users create contents for themselves and exchange them on social media. Social media include many kinds of forms, including weblog, wikis, videos and microblogs. One of the biggest characteristics of social media is *user-generated contents*.

Social media users often make posts about what happened around them: live performance, sports events and natural disaster, including earthquake. Figure 1 depicts the graph of tweet counts and the sizes of earthquake on March 11th 2011, the day of the Great Eastern Japan Earthquake. It is apparent that tweet counts and earthquake occurrences are correlated. It means that when earthquakes occurs, social media users make posts about those earthquakes.

Along with the popularization of social media, new methods for earthquake observation are appearing. These method use information about earthquakes posted on the internet by users. For example, the web site *Did You Feel It?*, operated by the United States Geological Survey (USGS), gathers earthquake information from web-site users through a

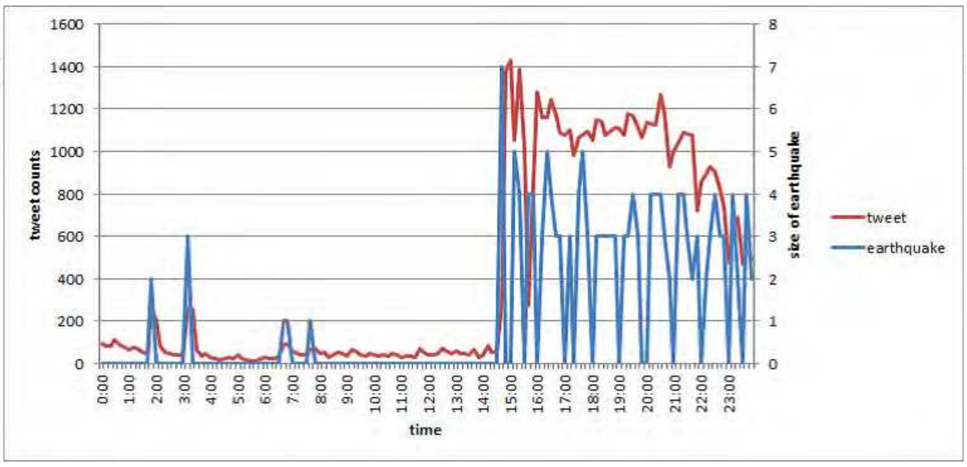


Fig. 1. Size of earthquakes and change of tweet counts on the day of the Great Eastern Japan Earthquake

questionnaire format(Intensity, 2005). From the Twitter web-site, *Toretter* extracts tweets that refer to earthquakes and estimates the location of an earthquake's epicenter using location information included with those tweets(Sakaki et al., 2010)

These methods treat social media users as sensors. We designate these virtual sensors as *social sensors*, which entail no costs. Unfortunately, such sensors provide a signal that is extremely noisy because users sometimes misunderstand phenomena, sleep, and are not near a computer.

We introduce these methods and explain a process for earthquake detection by analyzing social sensor information. We introduce current studies and services for earthquake observation using *social sensors* . Moreover, we explain *Toretter* as an example and describe its mechanisms.

2. Overview of earthquake observation by social sensors

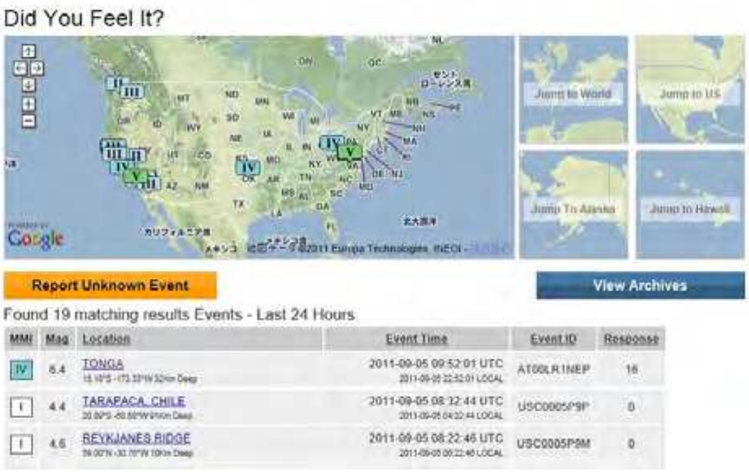
We explain the basic idea of *social sensors* and introduce internet service users as social sensors to observe earthquakes.

2.1 Earthquake observation services performed by social sensors

We introduce four earthquake observation services that use information from internet users. In this chapter, we examine Toretter as an example. We explain its detailed mechanisms in the next chapter.

Did You Feel It?

The web site *Did You Feel It?*, which is operated by United States Geological Survey (USGS), is shown in Fig. 2. Through the internet, it gathers earthquake information from users who experienced those earthquakes directly (Intensity, 2005).



Did You Feel It?

Found 19 matching results Events - Last 24 Hours

MMI	Mag	Location	Event Time	Event ID	Response
IV	6.4	TONGA 15.10°S -173.33°W 32km Deep	2011-09-05 09:52:01 UTC 2011-09-05 22:52:01 LOCAL	AT00LR1NEP	16
I	4.4	TARAPACA, CHILE 20.80°S -69.88°W 91km Deep	2011-09-05 08:32:44 UTC 2011-09-05 04:32:44 LOCAL	USC0005P9P	0
I	4.6	REYKJANES RIDGE 59.00°N -30.70°W 10km Deep	2011-09-05 08:22:46 UTC 2011-09-05 00:22:46 LOCAL	USC0005P9M	0

Fig. 2. Screenshot of *Did You Feel It*?

TED

The USGS also manages the Twitter Earthquake Detector (TED), which gathers tweets referring to earthquake occurrences from Twitter. They acquire location information and photographs attached to tweets and show this information related to maps(Survey, 2009).

iShake

The iShake project has developed a smartphone application (Fig. 3) that uses a phone to measure acceleration during an earthquake and report those data to researchers for processing (CITRIS, 2011). This project, conducted by UC Berkeley, is designed to create a system that moves beyond *Did You Feel It?*. Data from smartphone applications can complement data obtained from ground monitoring instruments, thereby improving the resolution and accuracy of earthquake intensity maps.

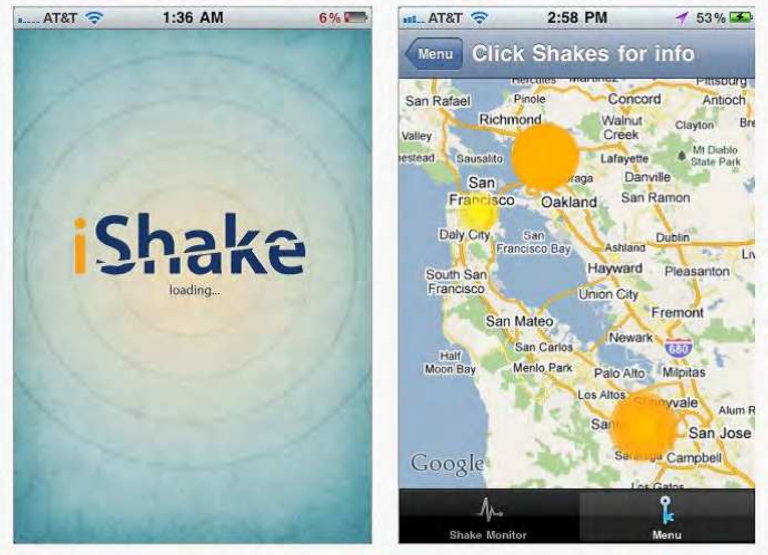


Fig. 3. Screenshot of *iShake*.

Toretter

Toretter extracts tweets referring to earthquakes and estimates the location of the earthquake epicenter using location information of those tweets(Sakaki et al., 2010). A temporal model and spatial model for earthquake detection are defined by social sensors. Then methods are proposed to detect earthquakes and to estimate the location of an earthquake epicenter automatically.

The Toretter mechanism is shown in Fig. 4.

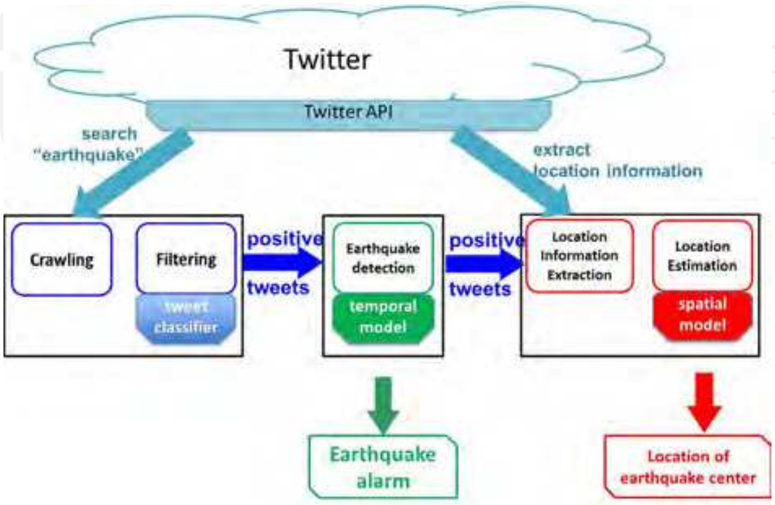


Fig. 4. Image of the Toretter mechanism.

First it collects tweets referring to earthquakes by crawling with the Twitter API and filtering the tweet messages using a tweet classifier. Second it tries to detect an earthquake from collected tweets based on a temporal model for earthquake detection. Finally, it extracts location information for each tweet from Twitter. The system uses that information and a particle filter to estimate the earthquake epicenter based on a spatial model for social sensors.

In this chapter, we explain methods of earthquake observation using social sensors according to the Toretter mechanism. We explain this entire process in the following section.

2.2 Overview of social sensors

We introduce the mode of *social sensors* and describe their features in comparison to physical sensors.

2.2.1 Basic idea of social sensors

Many methods and infrastructure can be used to observe events and natural phenomena using physical sensors: heavy traffic, air pollution, astronomical events, weather phenomena, and earthquakes are some examples. The basic mechanisms of such observations by physical sensors are presented on the right side of Fig. 5. First, a target event for observation occurs. Second, some sensors for the target event respond with a positive signal. Third, a central server collects signals from sensors and analyzes them. Finally, the server detects the target event or produces some observation values as output.

If users of social media observe an event, then similarly to physical sensors, they make posts about the event. For example, some Twitter users might post "Oh earthquake!" or "pouring rain, thunder & lightning " or "It's a double rainbow! & the moon is out. Beautiful!". These actions by users are analogous to the response of physical sensors to a stimulus: the users and sensors send a signal when an event occurs. Therefore, a user of social media is a sensor of a kind. We designate such sensors as social sensors.

An observation system incorporating social sensors is depicted on the left side of Fig. 5. First, an event occurs. Second, social media users make posts about the event. Third, the posts are collected at a central server and analyzed. Finally, the server detects the event or produces some observation value. This whole process corresponds to a process of observation by physical sensors, presented for comparison in Fig. 5

Methods for observing phenomena by physical sensors can be adapted to social sensors. Actually, some services based on social media use methods of observation resembling methods used with physical sensors.

Regarding Twitter users as social sensors, we can work with the following assumption.

1. Each Twitter user is regarded as a sensor. A sensor detects a target event and makes a report probabilistically.
2. Each tweet is associated with a time and location, which is a latitude–longitude pair.

2.2.2 Features of social sensors

Social sensors differ from physical sensors in some points. We describe features of social sensors in comparison to physical sensors.

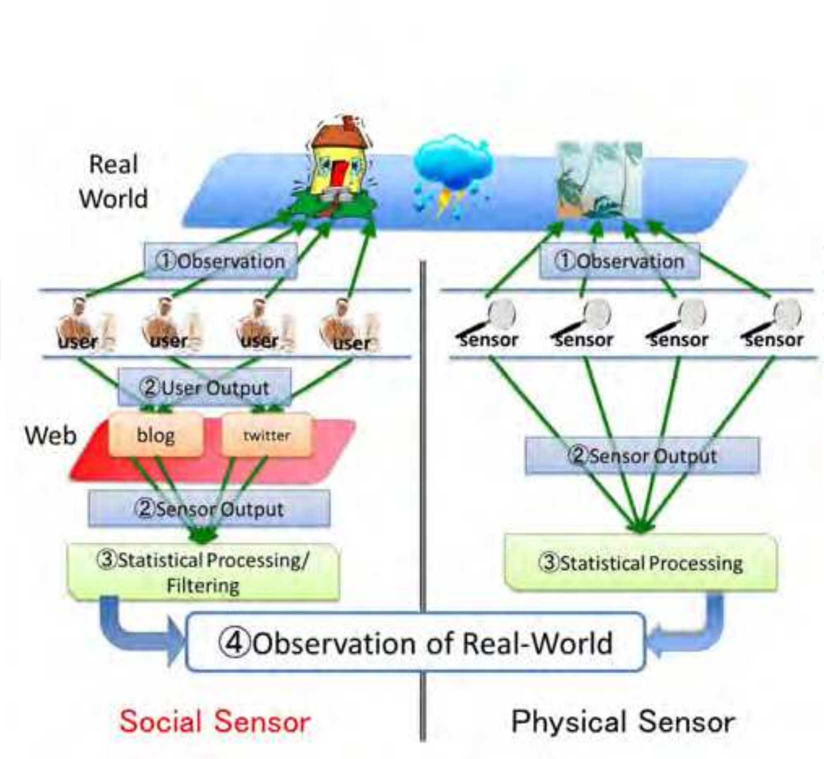


Fig. 5. Correspondence between event observation by social sensors and by physical sensors.

Social sensors are uncontrollable. They sometimes become inoperable because some users are not on-line; maybe they are sleeping or busy doing something else. They also function improperly more often than physical sensors because users misinterpret events more often than physical sensors. Therefore, it is necessary to know that social sensors are noisier than physical sensors and that their signals must be analyzed more carefully.

Social sensors, which are users of social media, are located over a wide area. They can give responses to events of many kinds, ranging from natural phenomena, such as earthquakes and hurricanes, to events related to human activities, such as heavy traffic, live performances, and elections. The extremely numerous social sensors all over the world present the possibility of responding to events of many kinds. In other words, detection of target events can be done with no cost to set up sensors. However, when using social media systems such as Twitter, which incorporate these social sensors, it is necessary to filter the signals (tweets) posted by social sensors (Twitter users) according to the event that is to be observed. Using some method, it is necessary to extract tweets referring to a target event. We summarize the features of social sensors and physical sensors in Table 1.

We explain these methods in the next section.

3. Tweet collection

In the first step portrayed in Fig. 4, it is necessary to collect tweets referring to an earthquake from Twitter. This process includes two steps: crawling tweets from Twitter and filtering out

features	physical	social
accuracy	high accuracy	noisy
versatility	target events only	event of any kind
cost	high	very low
processing	simple	complex

Table 1. Features of physical sensor and social sensors.

tweets that do not refer to the earthquake. For crawling and filtering tweets, we recommend using script programming languages, such as Python, PERL, and Ruby.

3.1 Crawling tweets from Twitter

To collect tweets or some user information from Twitter, one must use the Twitter Application Programmers Interface (API). Twitter API is a group of commands that are necessary to extract data from Twitter. Twitter has APIs of three kinds: Search API, REST API, and Streaming API. In this section, we introduce Search API and Streaming API, which are necessary to crawl tweets from Twitter. We explain REST API later because REST API is necessary to extract location information from Twitter information.

Additionally, it is known that Twitter API specifications are subject to change. When using Twitter API, it is necessary to know the latest details and requirements. They are obtainable from Twitter API documentation[1].

3.1.1 Twitter Search API

The Twitter Search API extracts tweets from Twitter, including search keywords or those fitting other retrieval conditions, in chronological order. It is possible to use language, date, location and other conditions as retrieval conditions.

When searching tweets including *earthquake* posted from 1 Aug 2011 to 5 Aug 2011, one might access the following URL:

http://search.twitter.com/search.json?q=earthquake&since=2011-09-01&until=2011-09-05

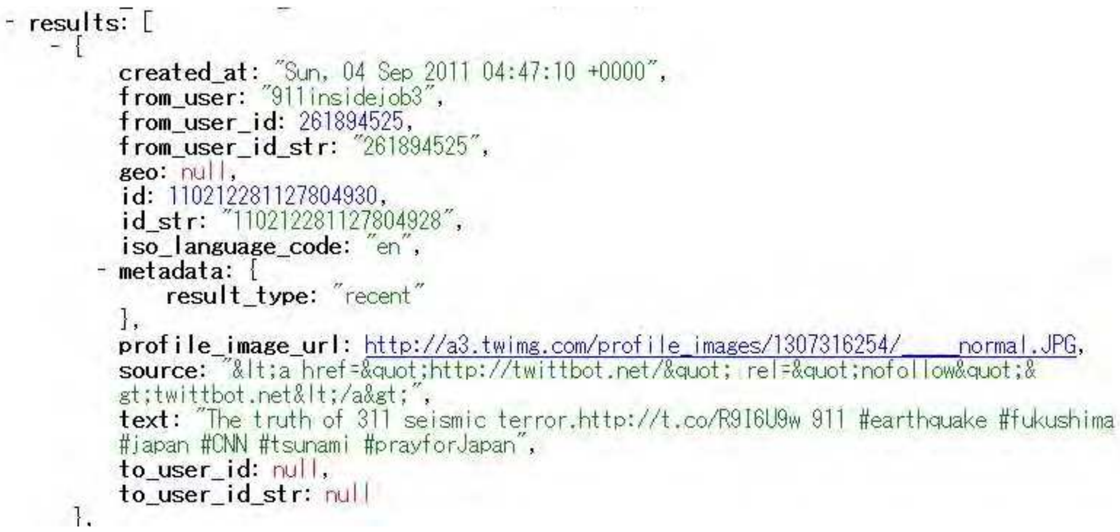
tweet time	user	tweet text
2011-09-04 04:47:10	user 1	The truth of 311 seismic terror.http://t.co/R9I6U9w 911 #earthquake #fukushima #japan #CNN #tsunami #prayforJapan
2011-09-04 04:47:09	user 2	FML! What did I say?! @..... RT @.... 24 HOUR EARTHQUAKE WARNING for San Diego, - 6.0+ likely - hey @....
2011-09-04 04:47:08	user 3	ML 2.3 SOUTHERN GREECE: Magnitude ML 2.3Region SOUTHERN GREECEDate time 2011-09-04 04:37:42.0 UTCLocation ...

Table 2. Search results of keyword *earthquake* after the conversion.

It is possible to obtain results in Fig. 6, as described in JavaScript Object Notation (JSON) format, which is a text-based open standard designed for human-readable data. It is possible to convert this result in Fig. 6 into Table 2 by parsing the result using a script programming language. Parameters that are often used to collect tweets are shown in Table 3 (This table is referred to Twitter API Documentation[2]).

[1] https://dev.twitter.com/docs

[2] https://dev.twitter.com/docs/api/1/get/search



```
- results: [
   - {
        created_at: "Sun, 04 Sep 2011 04:47:10 +0000",
        from_user: "911insidejob3",
        from_user_id: 261894525,
        from_user_id_str: "261894525",
        geo: null,
        id: 110212281127804930,
        id_str: "110212281127804928",
        iso_language_code: "en",
      - metadata: {
            result_type: "recent"
        },
        profile_image_url: http://a3.twimg.com/profile_images/1307316254/____normal.JPG,
        source: "&lt;a href=&quot;http://twittbot.net/&quot; rel=&quot;nofollow&quot;&
        gt;twittbot.net&lt;/a&gt;",
        text: "The truth of 311 seismic terror.http://t.co/R9I6U9w 911 #earthquake #fukushima
        #japan #CNN #tsunami #prayforJapan",
        to_user_id: null,
        to_user_id_str: null
    },
```

Fig. 6. Search results from Twitter Search API.

name	explanation	required	value
q	search keywords	required	-
rpp	the number of tweets to return per page	optional	up to 100
result type	search result of type	optional	mixed/recent/popular
until	tweets before the given date	optional	before today
since	tweets after the given date	optional	after 5 days ago
lang	restricts tweets to the given language	optional	jp/en/all/others

Table 3. Search conditions of Twitter Search API.

Some points must be considered when using Twitter Search API:

- It is possible to collect tweets posted only during the prior five days. It is not possible to search tweets posted six days ago.
- It is only possible to collect the latest 1500 tweets at one time.
  (Technically speaking, it is possible to access one page with a request and track pages back to the 15th page. One page includes 100 tweets at most. Therefore it is possible to acquire the latest 1500 tweets at one time.)
- One is limited to API requests.
  (No limit is published, but it is possible to access the Twitter Search API at least 500 times per hour.)

Therefore, we recommend the collection of tweets every 10 min or more often because it is impossible to crawl all tweets including *earthquake* if those tweets are posted at 2000 tweets per hour and one uses Twitter Search API every hour. Actually, tweets including *earthquake* were posted at more than 5000 per hour when the earthquake occurred on March 11, 2011.

Toretter requests the API command *search* 15 times every 5 min to collect the latest tweets each time: 180 command executions per hour.

3.1.2 Twitter Streaming API

The Twitter Streaming API extraction is defined in Twitter API documentation as follows:

> The Twitter Streaming API enables high-throughput near-real-time access to various subsets of public and protected Twitter data.

Twitter Streaming API provides some methods shown in Table 4, of which *filter* method can be used to crawl tweets related to earthquakes.

command	explanation
filter	returns public statuses that match one or more filtering conditions.
firehose	returns all public statuses. A few companies have permission to access this command.
link	returns all statuses containing http: and https:.
retweet	returns all retweets
sample	returns a random sample of all public statuses.(ratio is about 1%)

Table 4. Streaming API methods.

Filter method returns public statuses that match one or more filtering conditions. All conditions of *filter* are presented in Table 5. It is possible to use the parameter *track* to collect tweets because keywords can be set as a condition value of *track*.

command	explanation
follow	returns public statuses that reference the given set of users.
track	returns public statuses that include specified keywords.
locations	returns public statuses that posted from a specific set of bounding boxes to track.

Table 5. Conditions of *filter* methods.

When using a *filter* command with the parameter keyword, *earthquake*, it is necessary to create a file called *tracking* that contains *track=earthquake*. Then one can access the following URL:

https://stream.twitter.com/1/statuses/filter.json

Streaming API also returns results in the form of JSON, shown in Fig. 6. Therefore, it is possible to parse those results in the same way as results obtained with Search API.

It is possible to collect tweets including *earthquake* in real time. Some points must be considered when using Twitter Streaming API:

- The prepared server must have sufficiently high specifications to process all data received from Twitter.
- It is impossible to use some characters in Twitter Streaming API (e.g., Japanese characters can not be used in Twitter Streaming API).

Using Toretter, we want to detect earthquakes in Japan. For that purpose, it is necessary to collect tweets including *earthquake* in Japanese. However, Japanese characters cannot be used in Twitter Streaming API. Therefore, Toretter uses the Twitter Search API to crawl tweets. To collect tweets of languages other than English, it is necessary to check whether that language is supported by the Twitter Streaming API.

tweet	real-time
SYF News: Magnitude 7.0 earthquake shakes Vanuatu; no tsunami alert	no
HOLY **** EARTHQUAKE	yes
Powerful earthquake rocks Vanuatu, no tsunami warnings (Newkerala)	no
AAAAAAAAAH earthquake !	yes
Holy ****, that earthquake scared the **** outta me	yes
a year on after our very first earthquake... and the shakes are still happening	no

Table 6. Sample tweets and relevance of real-time earthquake detection.

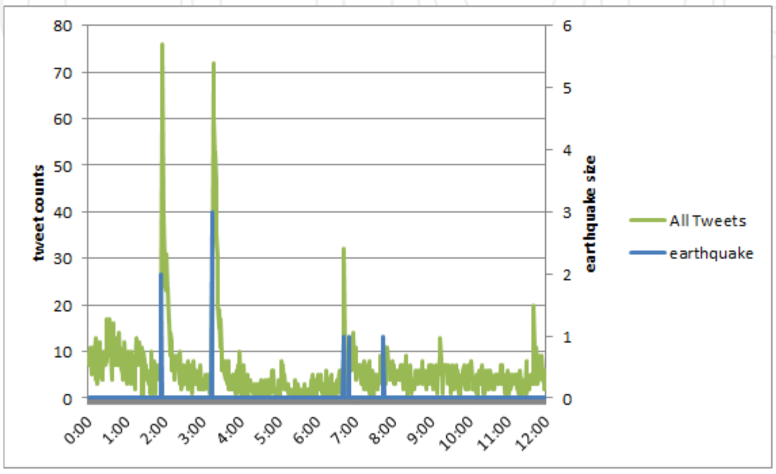


Fig. 7. Size of earthquakes and change of tweet counts on February 11, 2011

3.2 Filtering tweets using machine learning

We collected data from tweets including keywords related to earthquakes, such as *earthquake*, *shake*. Sample tweets are presented in Table6.

Those tweets include not only tweets that users posted immediately after they felt earthquakes, but also tweets that users posted shortly after they heard earthquake news, or perhaps they misinterpreted some sense of shaking from a large truck passing nearby. Figure 7 presents sizes of earthquakes and counts of Japanese tweets including the keyword *earthquake* on February 11, 2011. When the seismic activity reached its peak, the graph of tweets invariably showed a peak. However, when the graph of tweet counts showed a peak, the seismic activity did not necessarily show a peak. Some "false-positive" peaks of the graph of tweet counts arise from mistakes by users or some news related to earthquakes. Therefore, we must filter tweets to extract those posted immediately after the earthquake. We designate tweets posted by users who felt earthquakes as *positive* tweets, and other tweets as *negative* tweets.

Here, we describe the creation of a classifier to categorize crawled tweets into *positive* tweets and *negative* tweets, using Support Vector Machine: a supervised learning method.

3.2.1 Supervised learning

Supervised learning, a machine learning method, solves classification problem and regression problems analyzing training data. It is often used for spam mail filtering and gender estimation of Web users.

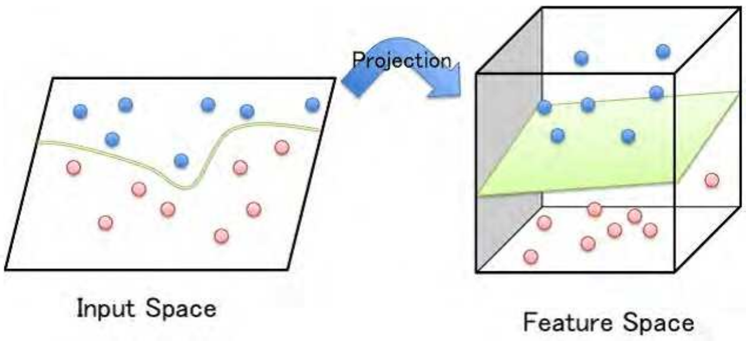


Fig. 8. Mechanism of Support Vector Machine.

Toretter uses Support Vector Machine (SVM), an extremely effective supervised learning method.

3.2.1.1 Support Vector Machine

SVM is a method used to create a classifier for two-class pattern classification. The SVM projects each training sample as points (as presented on the left side of Fig. 8) into multi-dimensional feature space. It creates a hyperplane that has the largest distance to the nearest training sample points of each class (as presented on the right side of Fig. 8). One must input positive samples and negative samples into SVM, which creates a classifier for two classes by searching the hyperplane.

To study them in detail, several books are useful (Bishop, 2006).

3.2.1.2 Process of creating a classifier using machine learning

Figure 9 depicts the process of supervised learning, which has three steps. We explain this process using an example of creation of a spam filter along the lines of Fig. 9

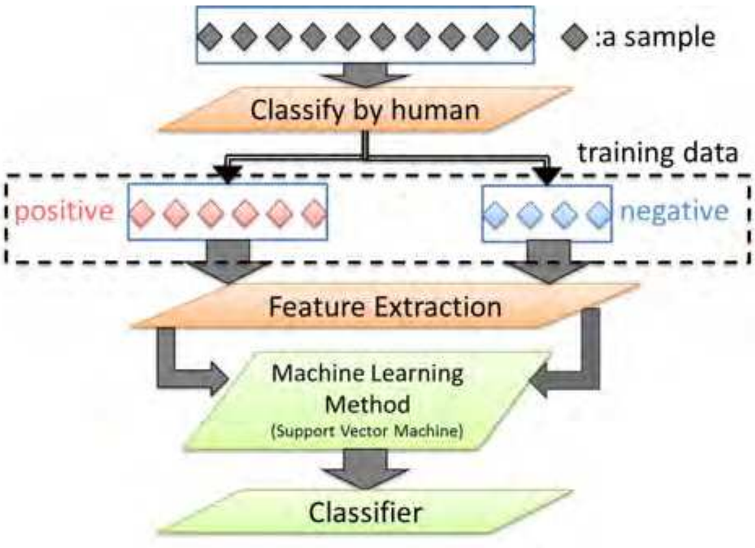


Fig. 9. Process of Machine Learning.

First, we prepare both sample collections of spam mails as positive samples and those of other mails as negative samples. Those must be classified manually by humans.

Second, we extract various features from samples. We must select effective features for classification. Effective features are those which positive samples seem to have and which negative samples do not seem to have, or vice versa. For example, all words included in samples are often used to create spam filters because we can infer that spam messages include words such as "Free!", "50% off!", and "Call now!" with high probability.

Third, we input both positive samples and negative samples with feature information and create a classifier for those samples. If inputting a new mail into the classifier, then it outputs a positive value or a negative value. If the output is positive, the new mail is regarded as a spam message.

3.2.2 Creation of sample data for the classifier

Positive samples and negative samples must be created manually. There are two points of consideration.

First, this process is very sensitive. One must classify positive tweets and negative tweets accurately. Therefore, it is necessary to acquire records of actual earthquakes. One must choose positive tweets referring to these earthquake records to classify them precisely.

Second, one must prepare equal numbers of positive tweets and negative tweets. The number of samples needed depends on the task. Generally, it is said that sample data must comprise 300–500 samples. Actually, one should increase the number of samples until finding the classification which provides sufficient performance.

3.2.3 Extraction of features from sample data

Next, one must select features of tweets for machine learning. In the spam mail filter example, words included in sample mails are chosen as features. Toretter uses features of three kinds. We explain them in detail and use the following sentence for explanation.

Oh! Earthquake happened right now!

Keyword features all words included in a tweet.
example sentence → Oh, earthquake, happened , right, now

Statistical features number of words in a tweet message and the position of the search keyword within a tweet
example sentence → number of words: *five*, the position of the search keyword: *second*

Context features words before and after a search keyword.
example sentence → *Oh*, *happened*

Statistical features are the most effective in these three features according to results of our earlier research(Sakaki et al., 2010). It is guessed that this is true because people who came across an earthquake were surprised and in an emergency situation so that they tend to post short tweets such as "Oh! earthquake!" and "It's shaking".

Of course, these features can differ depending on language, country, and culture. Therefore, effective features should be chosen when creating a filter for tweets.

feature ID	feature	feature ID	feature
0	I	1	am
2	in	3	Japan
4	earthquake	5	right
6	now		
7	*number of words in tweets*		
8	*position of search keyword*		
9	*word before keywords* is Japan		
10	*word after keywords* is right		

Table 7. Sample features for SVM-Light.

3.2.4 Applying machine learning

Some machine learning methods can create a classifier for any problem: Naive Bayes classifier, Neural Networks, Decision Tree, and Support Vector Machine. In this chapter, Support Vector Machine is used for our explanation because it is said that SVM is a superior method for classification problems and regression problems, and many SVM software packages exist. We treat SVM-Light, which is a popular SVM tool, as an example in this chapter.

Creating a classifier demands three steps.

3.2.4.1 Create training data from tweets

First, it necessary to convert tweet data into a training data file format for SVM-Light. The training data file format of SVM-Light is

```
    <target> <feature>:<value> <feature>:<value> ... <feature>:<value> # <info>
-1 1:0.43 3:0.12 9284:0.2 # abcdef
```

In this file format, each line corresponds to a single tweet. **<target>** expresses a polar of each tweet. $+1$ means positive and -1 means negative. **<feature>** expresses a feature ID of each feature and **<value>** expresses the weight of each feature in the tweet. Each feature should be assigned to each feature ID. For example, if one assigns each feature to each feature ID, as in Table 7, then a tweet conversion into a training data for SVM-Light as shown below.

I am in Japan, earthquake right now $\rightarrow$ +1 0:1 1:1 2:1 3:1 4:1 5:1 6:1 7:7 8:5 9:1 10:1

You must run the following command to create a classifier for tweets after converting positive tweets collected into a training data file *training data file*.

```
svm_learn ″training data file″ ″model file″
```

svm_learn is a command in SVM-Light to create a model file for classifier. After running *svm_learn*, it is possible to obtain *model file* as an output of *svm_learn*. It is possible to classify the tweet command *svm_classify* with this model file. When classifying new tweets into a positive class and negative class, each tweet is converted into *test data* in the same format as *training data*. Then the following command is executed.

```
svm_classify ″test data file″ ″model file″ ″output file
```

It is possible to obtain polars of each tweet in the *output file* New tweets are classifiable into a positive class and negative class by the classifier for tweets as described.

SVM-Light(Joachims, 2008), LIBSVM(Chih-Chung & Chih-Jen, 2011), and Classias(Okazaki, 2009) have compatibility such that the process we explain here is applicable to LIBSVM and Classias. (Toretter uses Classias for SVM tools.)

4. Earthquake detection from a time-series data using a probabilistic model

The second step of Fig. 4 detects an earthquake from positive tweets.

First, it is difficult to believe these tweets directly because some users misinterpret shaking caused by something other than an earthquake. Some ill-willed users post positive tweets to deceive others. This closely resembles physical sensors, and sometimes produces a wrong value. Therefore, we must process positive tweets to detect earthquakes with high accuracy, similarly to treating physical sensors.

Figure 10 depicts the sizes of earthquakes and counts of positive tweets filtered by SVM on Feb 11 2011. These two graphs are correlated: whenever an earthquake occurs, a peak appears in the graph of positive tweet counts. Therefore, we can detect earthquakes by detecting the peaks of positive tweet counts.

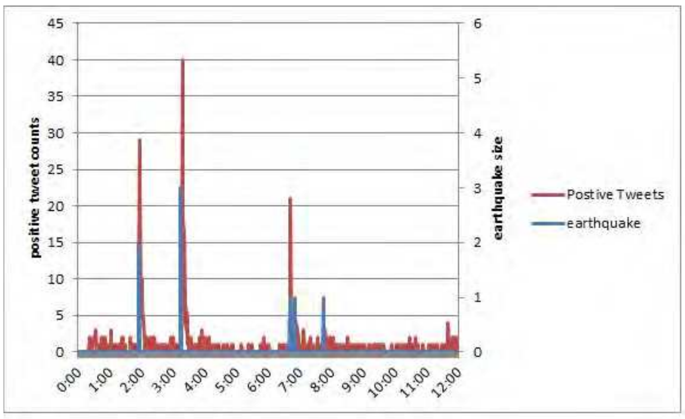


Fig. 10. Sizes of earthquakes and changes of filtered tweet counts Feb 11 2011.

Many methods have been used to detect peaks from time-series data for purposes such as burst detection(Kleinberg, 2002; Zhu & Shasha, 2003) and anomaly detection(Cheng et al., 2008; Krishnamurthy et al., 2003). Toretter uses a static rule *5 tweets in 5 min* that is calculated using an exponential function. We explain this method hereinafter.

4.1 Temporal model

To detect an earthquake using physical sensors, we must calculate the probability of earthquake occurrence based on signals from those sensors. Similarly, we must calculate the probability of earthquake occurrence from signals of social sensors. In this subsection, we explain the temporal model we use to calculate this probability.

Figure 11 presents graphs of positive tweet counts during earthquakes. In Fig. 11, the green line shows an exponential function. As shown here, the green line resembles the red line,

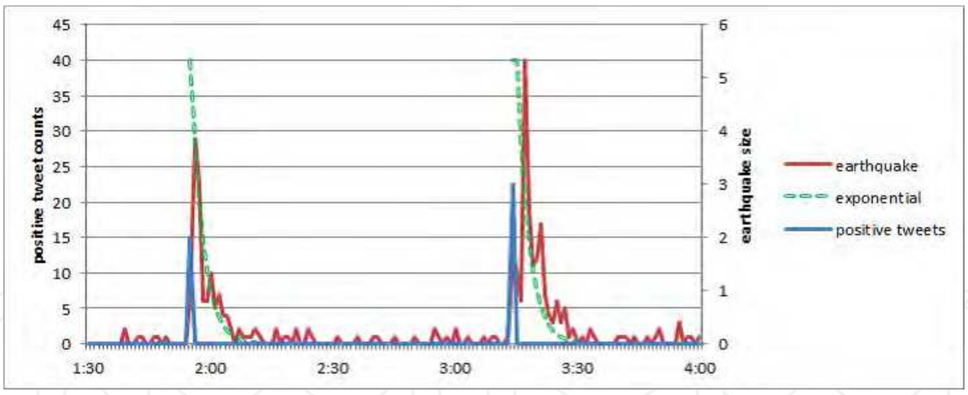


Fig. 11. Number of Tweets and Exponential Curve.

the graph of positive tweet counts. It can be inferred from these graphs that this frequency distribution of positive tweets is an exponential distribution, as expressed by the following equation(Sakaki et al., 2010).

$$f(t\lambda) = ke^{-\lambda t} \tag{1}$$

We express the number of sensors producing positive value at time t in $n(t)$. Here, $n(t)$ is equal to the number of positive tweets at time t. If n_0 sensors produce positive value at $t = 0$, then we can calculate the number of sensors for which the response is a positive value at time t using the following equation.

$$n(t) = n_0 \cdot e^{-\lambda t} \tag{2}$$

Therefore, we can calculate N_{t_a}, the number of sensors that produce a positive value from time 0 to time t_a, as presented below.

$$\begin{aligned} N_{t_a} &= \sum_{t=0}^{t_a} n(t) \\ &= n_0 \sum_{t=0}^{t_a} e^{-\lambda t} \\ &= n_0 \frac{1 - e^{-\lambda(t_a+1)}}{1 - e^{-\lambda}} \end{aligned} \tag{3}$$

We define the false-positive ratio of a sensor as p_f. In this case, we assume that we have n sensors and that all n sensors have the same false-positive ratio equally. The probability of all n sensors producing a false alarm is p_f^n. Therefore, the probability of earthquake occurrence can be estimated as

$$P(n) = 1 - p_f^n. \tag{4}$$

From Eq. 3, Eq. 4, we can calculate the probability of earthquake occurrence at time t_a.

$$\begin{aligned} p_{occur}(t) &= 1 - p_f^{N_{t_a}} \\ &= 1 - p_f^{n_0\left(1-e^{-\lambda(t_a+1)}\right)/\left(1-e^{-\lambda}\right)} \end{aligned} \tag{5}$$

4.2 Setup the condition for detection trigger

In the Toretter system, we detect an earthquake when *five positive tweets arrive in 5 min*, which means *five sensors produce positive signals in 5 min*. In this subsection, we explain how to determine this condition.

We set $\lambda = 0.34, p_f = 0.35$ (taken from our earlier research) to Equation (5) , by which we can calculate the probability of earthquake occurrence. When obtaining n_0 positive tweets, and given that we would like to make an alarm with false-positive ratio less than 1%, we can calculate t_{wait} as

$$t_{wait} = -\frac{1}{0.34} log\left(1 - \frac{1.264}{n_0}\right) - 1. \tag{6}$$

If we set $t_{wait} = 5$, then we can calculate $n_0 = 4.1$ from Eq. 6. Therefore, the trigger for earthquake detection is set as *five positive tweets come in 5 min* in Toretter. The trigger used for detection of earthquake calculation can be determined using an exponential function, as described in this section.

5. Location estimation from tweets

In this section, we explain a means to estimate the location of an earthquake epicenter by analyzing tweets. First, we introduce the kinds of location information to be acquired from tweets. Next, we explain methods to estimate the location of the earthquake epicenter.

5.1 Extracting location information from tweets

Two kinds of information are applicable for location estimation from tweets: using location information in the Twitter user profile or using *geotag* attached to tweets.

5.1.1 Location information in user profiles

The twitter user profile includes the location information of users. Of course, not all users make their location information public on the internet, but a sufficient number of users do so (This number varies among countries.).

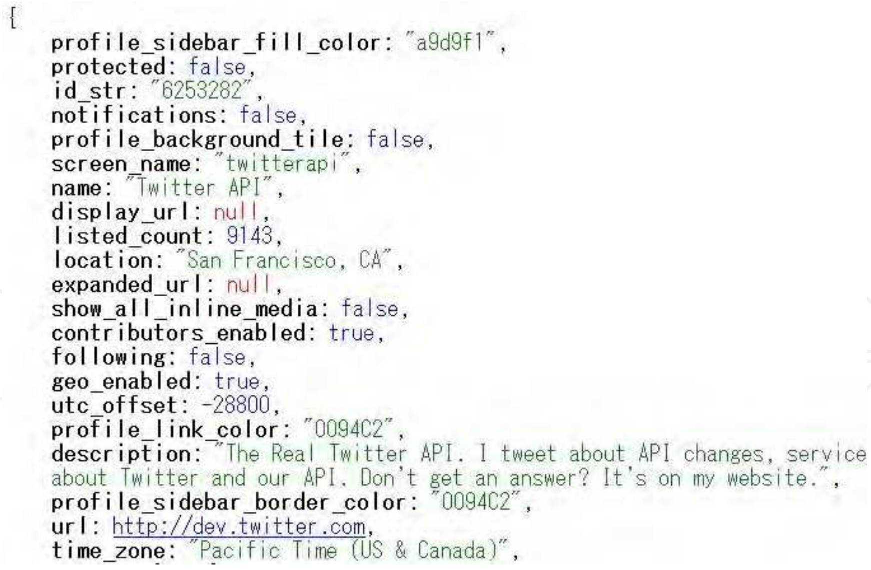
For earthquake detection, we collect positive tweets. We extract the location information of users who post those positive tweets for earthquake epicenter location estimation. Twitter REST API must be used to extract location information of users from Twitter.

Twitter REST API is one Twitter API included among all methods to use basic functions of Twitter. Many methods of using REST API exist. We use the *users/show* method to obtain user information. To extract user information of Twitter user *TwitterAPI*, it is necessary to access the following URL.

```
    http://api.twitter.com/1/users/show.json?screen_name=TwitterAPI
&include_entities=true
```

It is possible to obtain results in Fig. 12, which is described in JSON format, in the same manner as that used for Twitter Search API. It is possible to know from the result in Fig. 12 that Twitter user TwitterAPI resides in *San Francisco, CA*.

Some points to consider when using Twitter REST API are the following:



```
[
    profile_sidebar_fill_color: "a9d9f1",
    protected: false,
    id_str: "6253282",
    notifications: false,
    profile_background_tile: false,
    screen_name: "twitterapi",
    name: "Twitter API",
    display_url: null,
    listed_count: 9143,
    location: "San Francisco, CA",
    expanded_url: null,
    show_all_inline_media: false,
    contributors_enabled: true,
    following: false,
    geo_enabled: true,
    utc_offset: -28800,
    profile_link_color: "0094C2",
    description: "The Real Twitter API. I tweet about API changes, service
    about Twitter and our API. Don't get an answer? It's on my website.",
    profile_sidebar_border_color: "0094C2",
    url: http://dev.twitter.com,
    time_zone: "Pacific Time (US & Canada)",
```

Fig. 12. User information extraction from Twitter Search API.

- Some users do not register their location information, or register non-location data, such as *in a dream*, *anywhere*. Such non-location data should be ignored.
- API requests are limited.
  (The limit is published: it is possible to access the Twitter Search API about 150 times per hour without authorization.)

It is possible to access REST API 150 times per hour. This limit is sufficient to extract user information for location estimation of an earthquake epicenter because the earthquake-related tweets posted in the 5 min after an earthquake are most often fewer than 100. To expand the limit, one must register with Twitter and obtain an authorization called OAuth, according to the Twitter API Documentation[3].

Moreover one must convert location information acquired from Twitter into a latitude–longitude pair because human beings can understand places expressed by the names of places, such as *San Francisco*, but a computer can not understand where that place is. One must treat location information in the format of a latitude–longitude coordinate pair. At present, some web services can convert geographical names into a latitude–longitude coordinate pairs, such as the Google Maps API and Yahoo Maps API. Here we explain the Google Maps API.

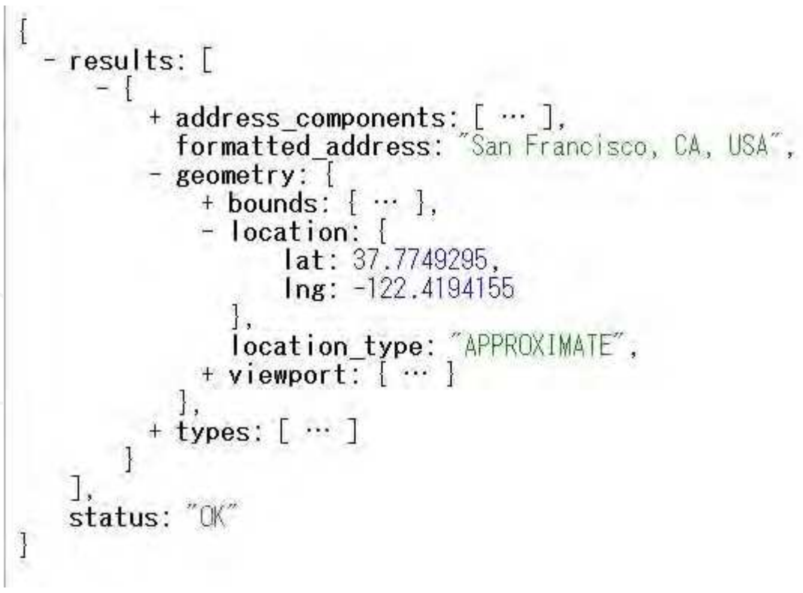
To convert *San Francisco* into a a latitude–longitude coordinate pair, one can access the following URL.

http://maps.google.com/maps/api/geocode/json?address=San
%20Francisco&sensor=false&language=en

Results are obtainable as in Fig. 13, which is described in JSON format, in the same manner as Twitter API. It is possible to convert *San Francisco* into $latitude = 37.7749295$, $longitude = -122.4194155$.

Location information related to an earthquake can be acquired as described above.

[3] https://dev.twitter.com/docs/auth



```
{
  - results: [
      - {
          + address_components: [ … ],
            formatted_address: "San Francisco, CA, USA",
          - geometry: {
              + bounds: { … },
              - location: {
                    lat: 37.7749295,
                    lng: -122.4194155
                },
                location_type: "APPROXIMATE",
              + viewport: { … }
            },
          + types: [ … ]
        }
    ],
    status: "OK"
}
```

Fig. 13. Result of geographical name converted using Google Maps API.

5.1.2 Geotags attached to each tweet

Some tweets have an attached geotag, which includes a latitude–longitude pair acquired from GPS. If positive tweets related to an earthquake include tweets with attached geotags, then it is possible to use these geotag data for location estimation. Geotag data can be extracted using the Twitter Search API. Therefore, GPS data can be obtained if stored when using crawl for those tweets by the Twitter Search API.

Geotag data are more accurate than location information of the Twitter user profile because they are acquired from GPS. Nevertheless, it is unusual that positive tweets referring to an earthquake include a sufficient number of tweets with attached geotags to estimate the earthquake epicenter location. Actually, a combination of location information of Twitter users and geotag should be used.

5.2 Location estimation using Bayesian filtering

If one can obtain sufficient location information from positive tweets, then estimating the location of the earthquake epicenter can be done using the information. Nevertheless, that information is often inaccurate. Alternatively if they are precise, then users might still be posting far from the earthquake epicenter. Therefore, it is preferred that the location of the earthquake epicenter be estimated probabilistically.

Several methods can be used to estimate the location of events from sensor readings using Bayesian Filters: Kalman filters, Multihypothesis tracking, Grid-based approaches, Topological approaches, and Particle filters.

We use particle filters as an example for explanation. Particle filters have high performance in belief, accuracy, robustness, and variety according to an evaluation by Fox et al. (Fox et al., 2003). Moreover particle filters work better to detect earthquakes from Twitter in the experiments by Sakaki et al. (Sakaki et al., 2010).

5.2.1 Spatial model

Each tweet is associated with a location. We describe a method that can estimate the location of an event from sensor readings. To define the problem of location estimation, we consider the evolution of the state sequence $\{x_t, t \in \mathbf{N}\}$ of a target, given

$$x_t = f_t(x_{t-1}, u_t), \;\; f_t : \mathcal{R}_t^n \times \mathcal{R}_t^n \to \mathcal{R}_t^n,$$

where f_t is a possibly nonlinear function of the state x_{t-1}. Furthermore, u_t is an i.i.d. process noise sequence. The objective of tracking is to estimate x_t recursively from measurements, as

$$z_t = h_t(x_t, n_t), \;\; h_t : \mathcal{R}_t^n \times \mathcal{R}_t^n \to \mathcal{R}_t^n,$$

where h_t is a possibly nonlinear function, and where n_t is an i.i.d. measurement noise sequence. From a Bayesian perspective, the tracking problem is to calculate, recursively, some degree of belief in the state x_t at time t, given data z_t up to time t.

Presuming that $p(x_{t-1}|z_{t-1})$ is available, the prediction stage uses the following equation.

$$p(x_t|z_{t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|z_{t-1})dx_{t-1}$$

Here we use a Markov process of order one. Therefore, we can assume that

$$p(x_t|x_{t-1}, z_{t-1}) = p(x_t|x_{t-1}).$$

In the update stage, Bayes' rule is applied as

$$p(x_t|z_t) = p(z_t|x_t)p(x_t|z_{t-1})/p(z_t|z_{t-1}),$$

where the normalizing constant is

$$p(z_t|z_{t-1}) = \int p(z_t|x_t)p(x_t|z_{t-1})dx_t.$$

To solve the problem, several methods of Bayesian filters are proposed such as Kalman filters, multi-hypothesis tracking, grid-based and topological approaches, and particle filters. For this study, we use particle filters, both of which are widely used in location estimation.

Additionally, we must consider the nonuniform distribution of Twitter users when we apply Bayesian filters to *social sensors* because *social sensors* are arranged non-uniformly to a greater degree than normal physical sensors are.

5.2.2 Location estimation using a particle filter

A particle filter is a Bayes filter that approximates a state probabilistically. It is a sequential Monte Carlo method. For location estimation, we maintain a probability distribution for the location estimation at time t, designated as the belief $Bel(x_t) = \{x_t^i, w_t^i\}, i = 1 \ldots n$. Each x_t^i is a discrete hypothesis related to the location of the object. The w_t^i are non-negative weights, called *importance factors*, which sum to one.

The Sequential Importance Sampling (SIS) algorithm is a Monte Carlo method that forms the basis for particle filters. The SIS algorithm consists of recursive propagation of the weights and support points as each measurement is received sequentially.

We use a more advanced algorithm with re-sampling. We use weight distribution $D_w(x, y)$, which is obtained from the Twitter user distribution to assess the biases of user locations[4] . The algorithm is shown as follows:

1. **Initialization**: Calculate the weight distribution $D_w(x, y)$ from Twitter users' geographic distribution in Japan.
2. **Generation**: Generate and weight a particle set, which means the N discrete hypothesis.
   (a) Generate a particle set
   $$S_0 = (s_0^0, s_0^1, s_0^2, \ldots, s_0^{N-1})$$
   and allocate them evenly on the map, as
   $$particle\ s_0^k = (x_0^k, y_0^k, w_0^k)$$
   $x, longitude; y, latitude; w, weight$
   (b) Weight them based on weight distribution $D_w(x, y)$.
3. **Re-sampling**
   (a) Re-sample N particles from a particle set S_t using weights of respective particles and allocate them on the map. We allow re-sampling of more than that of the same particles.
   (b) Generate a new particle set S_{t+1} and weight them based on weight distribution $D_w(x, y)$.
4. **Prediction**: Predict the next state of a particle set S_t from Newton's motion equation.
   $$\begin{aligned}(x_t^k, y_t^k) &= (x_{t-1}^k + v_{x_{t-1}}\Delta t + \frac{a_{x_{t-1}}}{2}\Delta t^2, \\ &\qquad y_{t-1}^k + v_{y_{t-1}}\Delta t + \frac{a_{y_{t-1}}}{2}\Delta t^2) \\ (v_{x_t}, v_{y_t}) &= (v_{x_{t-1}} + a_{x_{t-1}}, v_{y_{t-1}}, a_{y_{t-1}}) \\ a_{x_t} &= \mathcal{N}(0; \sigma^2),\ \ a_{y_t} = \mathcal{N}(0; \sigma^2).\end{aligned}$$
5. **Weighing**: Re-calculate the weight of S_t by measurement $m(m_x, m_y)$ as follows.
   $$\begin{aligned}dx_t^k &= m_x - x_t^k,\ \ dy_t^k = m_y - y_t^k \\ w_t^k &= D_w(x_t^k, y_t^k) \cdot \frac{1}{(\sqrt{2\pi}\sigma)} \\ &\qquad \cdot exp\left(-\frac{(dx_t^{k2} + dy_t^{k2})}{2\sigma^2}\right)\end{aligned}$$
6. **Measurement**: Calculate the current object location $o(x_t, y_t)$ by the average of $s(x_t, y_t) \in S_t$.
7. **Iteration**: Iterate Steps 3, 4, 5, and 6 until convergence.

[4] We sample tweets associated with locations and obtain a user distribution that is proportional to the number of tweets in each region.

6. Evaluation and application

In this section, we explain how to evaluate results of experiments and describe points that should be considered when applying these methods.

6.1 Selection of the target area

Three conditions must be met to apply methods for earthquake observation from social media.

The first is that a sufficient number of people use Twitter in a targeted area. The second one is that several earthquakes occur each year for a target area. The third one is that infrastructure should be set up in a target area.

These three conditions are needed in each step of earthquake detection and location estimation. A sufficient number of tweets and a certain number of earthquakes are needed to create a classifier for tweets and to estimate the locations of earthquake epicenters. Accurate logs of earthquakes are also necessary to calculate the false-alarm probability of social sensors and to evaluate the earthquake detection system performance.

If creating a classifier and setting a trigger for earthquake detection in an area and applying them in another area, then the third condition is not indispensable. However, the first condition and the second condition are necessary in both areas.

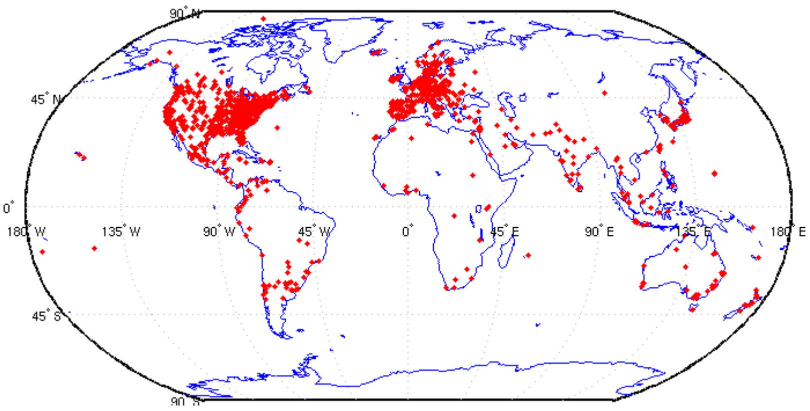


Fig. 14. Twitter user map.

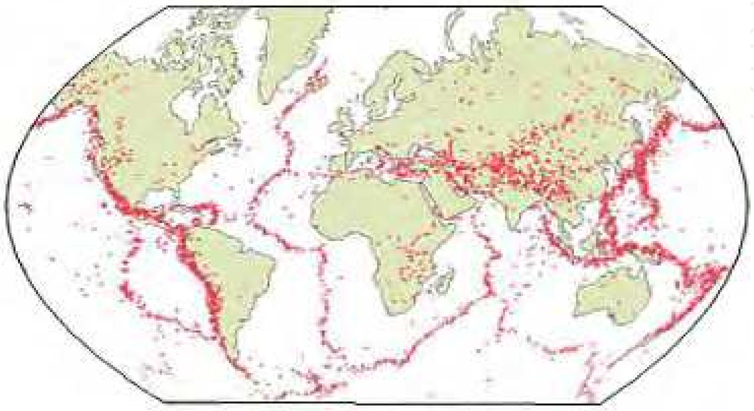


Fig. 15. Earthquake map.

Figure 14 depicts the Twitter user distribution map and Fig. 15 depicts an earthquake occurrence distribution map. Earthquake detection using information from Twitter users is applicable in overlapping areas of these two maps: for example, Japan, the west coast of the U.S., Indonesia, Turkey, Iran, and Italy.

The number of Twitter users has been increasing continuously. Therefore, those areas can probably be expanded. Additionally, if one uses social media other than Twitter, then overlapping areas might be changed.

Therefore, a target area should be chosen very carefully to apply the methods described in this chapter.

6.2 Evaluation of earthquake detection

To evaluate the performance of earthquake detection and earthquake epicenter location estimation, one must collect earthquake data from some organizations. Those data must include information about an approximate time point of an earthquake and approximate position of an earthquake epicenter. Moreover, it is better that they include the exact time of an earthquake, the longitude and latitude of an earthquake epicenter, and the seismic intensity of earthquakes in each region.

For example, the Japan Meteorology Agency (JMA) publishes an earthquake database on the Web, which includes a time, magnitude, and earthquake intensities at each point of area, a place of earthquake epicenter of all earthquakes above level 1 on the Japanese seismic intensity scale[5]. The USGS publishes similar data on the Web[6].

Data of such kinds can be obtained by crawling. They can be used to create training data for classifiers and to evaluate the performance an earthquake detection system.

7. Conclusion

Our research is an early approach to using Twitter as a social sensor for earthquake observations. It is meaningful that we apply methods by ordinary physical sensors to earthquake detection by social sensors. Furthermore, we present the possibility of earthquake detection without installing numerous physical sensors. The method is effective for earthquake observations in some countries where a few seismic sensors exist. However, it is difficult to detect earthquakes occurring in oceanic areas or less populated areas using methods we introduced in this chapter. Therefore, we must verify that earthquake detection by social sensors is effective when we apply these methods. Furthermore, the applicable scope of the earthquake observation by social sensors can be extended considering a stochastic gradient, more detailed probabilistic models, and so on. Many subjects remain to be explored in future work.

8. References

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*, Vol. 4 of *Information science and statistics*, Springer.

[5] http://www.seisvol.kishou.go.jp/eq/shindo_db/shindo_index.html

[6] http://neic.usgs.gov/neis/qed/

Cheng, H., Tan, P.-N., Potter, C. & Klooster, S. (2008). Data mining for visual exploration and detection of ecosystem disturbances, *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems - GIS '08* p. 1.

Chih-Chung, C. & Chih-Jen, L. (2011). LIBSVM-a Library for Support Vector Machine. URL: *http://www.csie.ntu.edu.tw/ cjlin/libsvm/*

CITRIS (2011). iShake -Mobile Phones as Seismic Sensors-. URL: *http://ishakeberkeley.appspot.com/*

Fox, D., Hightower, J., Schulz, D. & Borriello, G. (2003). Bayesian filtering for location estimation, *IEEE Pervasive Computing* 2(3): 24–33.

Intensity, M. (2005). Did You Feel It ? Citizens Contribute to Earthquake Science, *Technical Report March*, U.S. Geological Survey. URL: *http://earthquake.usgs.gov/earthquakes/dyfi/*

Joachims, T. (2008). SVM-Light. URL: *http://svmlight.joachims.org/*

Kleinberg, J. (2002). *Bursty and hierarchical structure in streams*, ACM Press, New York, New York, USA.

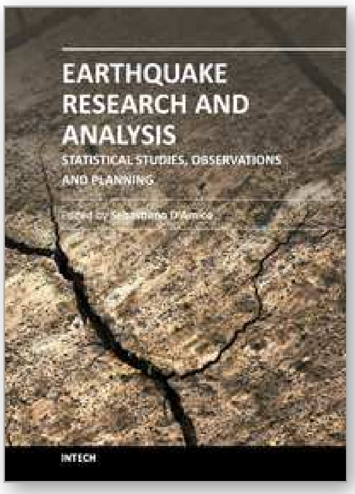
Krishnamurthy, B., Sen, S., Zhang, Y. & Chen, Y. (2003). Sketch-based change detection: methods, evaluation, and applications, *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*, ACM, pp. 234–247.

Okazaki, N. (2009). Classias. URL: *http://www.chokkan.org/software/classias/index.html.en*

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

Survey, U. S. G. (2009). Twitter Earthquake Detector (TED). URL: *http://recovery.doi.gov/press/us-geological-survey-twitter-earthquake-detector-ted/*

Zhu, Y. & Shasha, D. (2003). Efficient elastic burst detection in data streams, *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '03*, ACM Press, New York, New York, USA, p. 336.


EARTHQUAKE RESEARCH AND ANALYSIS
STATISTICAL STUDIES, OBSERVATIONS AND PLANNING
Edited by Sebastiano D'Amico
INTECH

Earthquake Research and Analysis - Statistical Studies, Observations and Planning
Edited by Dr Sebastiano D'Amico

ISBN 978-953-51-0134-5
Hard cover, 460 pages
Publisher InTech
Published online 02, March, 2012
Published in print edition March, 2012

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821