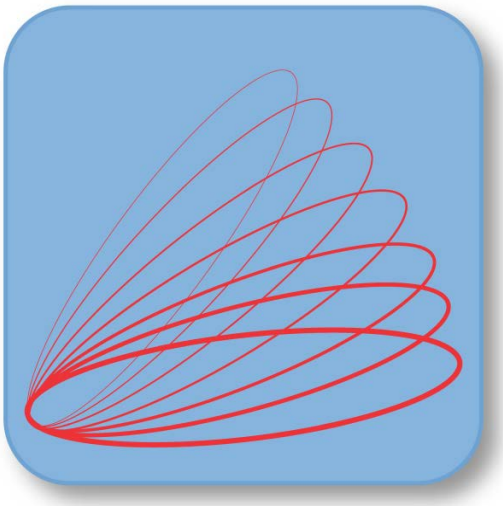
Centre for Geo-Information

Thesis Report GIRS-2014-19

Detecting Spatial-Temporal Events Based on Twitter Data using Named Entity Recognition

LIU Yue

2014/04/01



WAGENINGEN UNIVERSITY
WAGENINGEN UR

Detecting Spatial-Temporal Events Based on Twitter Data using Named Entity Recognition

LIU Yue

Registration number 890112523040

Supervisors:

Dr. Ir. Arend LIGTENBERG

A thesis submitted in partial fulfilment of the degree of Master of Science
at Wageningen University and Research Centre,
The Netherlands.

1st April 2014
Wageningen, the Netherlands

Thesis code number: GRS-80436
Thesis Report: GIRS-2014 -19
Wageningen University and Research Centre

Laboratory of Geo-Information Science and Remote Sensing

For David Aguirre Dávila, in memoriam.

ACKNOWLEDGEMENT

First of all, I would like to express many thanks to my supervisor, Dr. Ir. Arend Ligtenberg, who guided me on this thesis with his solid knowledge, abundant experience and warm-heart to young student.

Then, many thanks to my parents, for their financial and mental supports.

I also want to send my heartfelt thanks to my designer friend AN Xinzhuo from the University of Washington, who designed the picture on the front cover of this report.

My special salute and gratitude would be addressed to all the people who made their contributions to the open source community on developing and maintaining the software and extend packages which were used in this thesis. The notion of open source is great and selfless.

At last, thanks for accompanying and supporting from PhD candidate YIN Fang, who is the one in my world. And also thanks to other friends in Netherlands, United States and China.

LIU Yue

At Wageningen, the Netherlands

ABSTRACT

As becoming popular in our daily life, social network services provide a huge collection of volunteer data which can contribute to different kinds of research and inspire the scientists to think about achieving data in new perspective. In this thesis, an experiment was executed to apply social network data on the field of Geo-information sciences.

This thesis aimed to detect Spatial-Temporal Events from people's tweeting behaviour. A raw tweets dataset, which contained the text and time of about fifteen million tweets, was caught from Twitter streaming as the data resource of this research. Different from traditional geo-targeting methods, the spatial factor of tweets was achieved by Named Entity Recognition and Geo-coding. The Stanford Named Entity Recognizer model was used to extract location names from text of tweets, which was proven to provide enough geography information for detecting Spatial-Temporal Events. The Geo-coding was then applied to order location names and hierarchize them with administrative levels. This thesis pointed out possible inaccuracy produced by Stanford Named Entity Recognizer model, yet the validation was not applied since the limit of time and human resource.

The tweets, after location name extracting and geo-coding, entered a time-series decomposition and seasonal adjustment model based on local regression, then five patterns of interests were detected by manually checking with assisting from statistical method. Three Spatial-Temporal Events were detected from these five patterns. The online news was used to prove that these three Spatial-Temporal Events were correctly existed at the time and location we detected. As an experiment, this thesis successfully detected Spatial-Temporal Events from raw tweets data and pointed out the way of developing this topic in future.

Keywords: Stanford Named Entity Recognizer; Detecting Spatial-Temporal Events; Time-series decomposing; Tweets data.

TABLE OF CONTENTS

TABLE OF TABLES

TABLE OF FIGURES

CHAPTER 1: INTRODUCTION

In the introduction chapter, the research topic of this thesis will be introduced, including the context and background of this thesis, the definition of existing problems which lead to the objectives and questions of this research.

1.1 CONTEXT AND BACKGROUND

Twitter[1], as a microblogging service, has become worldwide popular in recent years. It allows users to publish real-time textual messages (called "tweets") which can be read by family members, friends and other interested observers. Mass textual tweets are generated by worldwide users via Twitter every day, in different languages, from different locations tweeting about different aspects of daily life(Lunden, 2012; Twitter, 2011).

From the perspective of public, Twitter, which is known as one of the most famous microblogging services around the world, has been making its contributions to the daily communications of people since it was founded. When it comes to scientists' point of view, Twitter also offers huge real-time volunteer datasets generated by a large numbers of users coming from different countries, with different backgrounds, and speaking different languages. The curiousness of applying this dataset on Geo-information studies drove me to develop my knowledge in this field and finish this report of my Master degree thesis.

1.2 PROBLEM DEFINITION

Lots of research was done based on of Twitter. One genre of studies on Twitter focused on tweets, users and other aspects of Twitter itself, for instance, three perspectives of user's influence in twitter was measured at 2010, concluded that influential users are more predictable than theory indicated(Cha, Haddadi, Benevenuto, & Gummadi, 2010). Another genre focused on regarding tweets data as a data resource for others researches, for example, Chew and Eysenbach analysed tweets related to the H1N1 outbreak at 2009, pointed out that the tweets data could be a source for health authorities to be aware of the public(Chew & Eysenbach, 2010). Vieweg, Hughes, Starbird and Palen analysed tweeting behaviour during two hazards in American, explored the features of Twitter information during the emergencies (Vieweg, Hughes, Starbird, & Palen, 2010).

With the development of geo-targeting function of Twitter, tweets began to contribute to studies in the field of Geo-information as a spatial-temporal data resource. The Geo-targeting, which describes the spatial characteristic, and the real-time temporal characteristic of tweets are both focused in these studies, for instance, White and Roth introduced TwitterHitter, an Geovisual application based on geo-referenced tweets, one example was described about the contribution from TwitterHitter to the crime analysis(White & Roth, 2010). Okazaki and Matsuo established a real-time event notification system of earthquakes from geo-referenced tweets(Okazaki & Matsuo, 2011).

These two examples studied the detecting and analysing of some typical Spatial-Temporal events based on Twitter data, which are criminal acts and earthquakes. However, these research still highly rely on the geo-targeting service provided by Twitter itself. Once the geo-targeting is disabled by user (it is set to disable as default on Twitter), it will be hard for them to extract geo-information in order to detect earthquakes, criminal acts, or other types of Spatial-Temporal events. Recent research revealed that only 0.77% of tweets had Geo-location information in 2012 (Lunden, 2012).

[1] www.twitter.com

Furthermore, geo-targeting presents the location where the user posted the tweets, which is suitable for the events happened physically close to the user, for instance, the traffic jam, earthquake and crime. In terms of the events which are remote to the current location of the user, the capability of geo-targeting may be limited, for example, a tweet from a user in Amsterdam who talks about the "Oktoberfest" in Munich, the important location "Munich" will be missed by geo-targeting using the direct location provided by Twitter, which will only offer one location "Amsterdam". The location information of tweet should contain both the place it was posted and the place it was talking about. The locations which the tweets are talking about are hidden into the text, so that they are called the "latent location information", where the "latent" means the location name appears in the text of tweets.

Comparing to geo-targeting, it is not easy to extract the latent location information of tweet. Here we have to face a task to identify and extract location information from a short textual message (tweet) by computer. Rau presented a system to recognize and extract company names from text in 1991(Rau, 1991), which was known as the first research about recognizing the names (company name) from text(Nadeau & Sekine, 2007). At the Sixth Message Understanding Conference (MUC-6)(Sundheim, Road, Diego, Grishman, & York, n.d.), the term "Named Entity" was created to refer to the information units like names (person, organisation and location names) and numeric expressions (time, date, money and per cent expressions). The Named Entity Recognition (NER) aims to locate and classify the "Named Entity" from sentences automatically. Fleischman attempted to use NER to identify location names from sentences and classify those location names into different categories in 2001(Fleischman, 2001). With the development of NER, "location" had become a typical basic class of named entities(Nadeau & Sekine, 2007).

The initial NER models were based on the handcrafted grammar rules which were precise but costly. In recent years, the machine learning approach was used in the NER field, unsupervised, semi-supervised and supervised learning methods were applied on the training dataset to recognise the named entity based on the statistical models. In terms of NER for tweets, recent research pointed out that the traditional methods may meet challenges when they face the tweets which lack context or have abbreviations(Ritter, Clark, & Etzioni, 2011). Particular NER model for tweets has become a new interest and a few achievements had been described(Finin, Murnane, & Karandikar, 2010; Liu, Zhang, Wei, & Zhou, 2011; Locke, Martin, & Ph, 2009; Ritter et al., 2011). It seems that the NER could fit our task of extracting location name from texts, but more experiments and practises need to be done.

Another limitation of current Twitter& Spatial-Temporal Events research is that most of them are only deal with keyword search of a given event (Okazaki & Matsuo, 2011; Tr-, Labs, Zhao, & Zhong, 2011). These keywords need to be defined by a human beforehand. It will be useful if the detection of events could be applied without a predetermined keyword, especially for some emergency issues. Mathioudakis and Koudas described a monitor system on tweets to detect the trend of tweets and summarize the real-time hot topics(Mathioudakis & Koudas, 2010). Cataldi, Caro and Schifanella also developed a technique to detect emergent topics from twitter(Cataldi, Caro, & Schifanella, 2010). These methods which tried to achieve the keyword automatically relied on statistics of the existed texts of tweets. It will give a new vision if the events would be identified based on the location information of tweets, not the text of them.

My thesis will attempt to realize the automatic detection of spatial-temporal events related to one location based on NER. I will try to apply a proper NER model on tweets to extract location information hidden in the text, then develop a clear definition of tweets-based spatial-temporal events. This clear definition should include basic characteristics of event, such as

types and structure of spatial-temporal events. In this way, an emergent event, such as a large accident, will be detected by the rapid appearance of new tweets posted from this location or talking about this location.

1.3 RESEARCH OBJECTIVES AND RESEARCH QUESTIONS

1.3.1 Research Objectives

The objective of this thesis is to develop and demonstrate a method and technique to detect spatial-temporal events based on NER approach.

Sub objectives

- Extract latent location information from the text of tweets with the help of an existing NER application.
- Collect the tweets data with latent location information in a clear data-structure.
- Define spatial-temporal events including the characteristics and classifications of it.
- Transfer the characteristics, classifications and structure of spatial-temporal events from theories into an algorithm which can be used in a prototype to detect the spatial-temporal events from an example dataset of tweets.

1.3.2 Research Questions

- How to define Spatial-Temporal Events? Can Spatial-Temporal Events be defined mathematically?
- What NER based methods are suitable to extract the latent location names from the text of tweets?
- How to translate the location names to geographic coordinates necessary for further analysing?
- Which characteristics of tweeting behaviour can help to identify spatial-temporal events?
- How to detect Spatial-Temporal Events from these characteristics?
- How to validate the effect of NER and the Spatial-Temporal Events detecting model?

CHAPTER 2: LITERATURE REVIEW

Although there is not a widely accepted definition of spatial-temporal events, the occurrence of a spatial-temporal events could be described as a phenomenon which is caused by the change of time and location, for example a two hour long concert could be seen as an event with a changed time and unchanged location; an airplane flying on its route could be seen as an event with both changed time and changed location. Based on the definition of the three elements of basic events: actor, time and location, Lauw and *et al* defined the spatial-temporal event as a subset of basic events, while the subset was selected via a few conditions on these elements respectively(Lauw, Lim, Pang, & Tan, 2005).

The detection of a Spatial-Temporal Event has been applied on several fields. For these research, the type of data resource is an important fundamental to distinguish them. Lots of research about Spatial-Temporal events based on the image and video sequence data as data resource. For example, Piotr Dollar and et al. tried to recognise visual behaviours via spatial-temporal features(Dollár, Rabaud, Cottrell, & Belongie, 2005); Niebles, J.C. and et al developed an approach for detecting and monitoring human actions by spatio-temporal words and space-time interests points on image/video(Niebles, Wang, & Fei-Fei, 2008). These research mainly used sequence of video/image as data resource. Their scales were limited within the range of visibility that impeded their extending on geographical research. At Geographical scale, the spatial factor (location) of data source is more difficult than temporal factor (time) to be collected. The "sensor" is needed for collecting spatial factor. Recent related research about the detection of spatial-temporal events focused on using a given type of sensor to collect location information of a spatial-temporal event(Lauw et al., 2005; Yin, Hu, & Yang, 2009).

For a small area, like a room or a field, there could be a dedicated sensor network to collect the spatial-temporal data. For instance, Yin, Hu and Yang developed a new algorithm on detecting spatial-temporal events based on the spatial-temporal data obtained from motes and light strength sensor of a sensor network (Yin et al., 2009). For a larger size, in another word, a geographical size ranged from community-level to national-level, regardless of the varied spatial-temporal data sources, the GPS (Global Positioning System) sensor and mobile/stable network played an important role for mining spatial-temporal data. For example, Lauw and et al mined the spatial-temporal events based on a dataset obtained from the internet domain at a campus-size sampling area (Lauw et al., 2005), Kisilevich and et al (Kisilevich, Krstajic, Keim, Andrienko, & Andrienko, 2010) analysed the activity and behaviour of people based on the spatial-temporal events detected by the geo-targeted photos from Flickr [1] and Panoramio[2], while the geo-targeting service was powered by the GPS service. As similar as other data source, when the Twitter was selected for research on a spatial-temporal events, the research relied on the GPS sensor as well (Okazaki & Matsuo, 2011; White & Roth, 2010).

For most of the research tried to use social network information as data resource, the geo-targeting became the essential function to extract spatial factor. Although the geo-targeting is supported on most of the popular social network service platforms and it is also supported by the smartphone to provide geo-targeted photos, only few user enable geo-targeting ult on their smartphone. This caused lots of social network data failed to enter the event detection models because they were not geo-targeted. As it has been mentioned in the problem definition, there were only 0.77% tweets geo-targeted in 2012 (Lunden, 2012). An innovation is needed to allow all tweets data entering the event detection models for avoiding this kind of wastages of

[1] www.flickr.com
[2] www.panoramio.com

data. Lee explored the density-based clustering approach on tweets to detect spatial-temporal events, which produced good results (Lee, 2012). His method inspired me because it tried to extract location information without the help of geo-targeting. It is called semantic analysing approach which aims to understand human's language by computer and figure out valuable information from the text of language.

The NER, as one of the semantic analysing methods fitting this research, scans all the words in a sentence and put them into pre-defined classes. From the very beginning of the development of NER methods, "location" has been a default type as pre-defined class (Nadeau & Sekine, 2007). Even though not all social network data is geo-targeted, all of them have texts. The advantage of NER model is that it could analyse the texts of social network data and detect locations from them, which solve the losing of data caused by un-geo-targeting.

The concept and approach of NER have been applied on analysis of social network platforms. Xiaohua Liu and et al. combined popular NER methods to be a new model then trained it on tweets, which produced a good result (Liu et al., 2011). Fabian Abel and et al. developed a system using several semantic analysis method including NER to detect incidents from tweets(Abel, Hauff, Houben, Stronkman, & Tao, 2012). However, as a trend of extracting locations from social network data, there is still further research to be done on locations in tweets resulted from NER. So in this thesis, I proposed a new train of thought on detecting spatial-temporal events from Twitter, which aimed to obtain the location information without the GPS or other sensors. This thesis tried to study whether the location information could be mined from the text of tweets directly or not.

To detect events from the data in time sequence after receiving the spatial factor by NER, time-series analysis is a useful method to assist us to detect pattern of interests of spatial-temporal data. Guralnik and et al. develop a mature algorithm of change-point detection for detecting events from time-series data (Guralnik & Srivastava, 1999); In the research of Sakaki and et al. which detected earthquakes from tweets, the time-series was also used for detecting the starting time point of earthquakes (Sakaki, Okazaki, & Matsuo, 2010).

With the assistance from NER and time-series analysis, this thesis aimed to collect tweets data and extract spatial factor from them, then detect Spatial-Temporal Events from these tweets semi-automatically.

CHAPTER 3: METHODOLOGY

3.1 CONCEPTUAL MODEL

This thesis aimed to develop a model consisting of different modules for detecting spatial-temporal events. Both the spatial factor as well as the temporal factor of the data resource should be collected for detecting.

In general, the detection of a Spatial-Temporal Event contains three steps: 1) acquire the original data, 2) convert original data to the spatial-temporal data, and 3) detect event from the spatial-temporal data. As there was not off-the-shelf dataset available, this model began with the collecting of tweets. And the key point of step 2 is to extract spatial factors and embed these factors on correct positions, so in practice, the overview design of this model was shown in Figure 1:

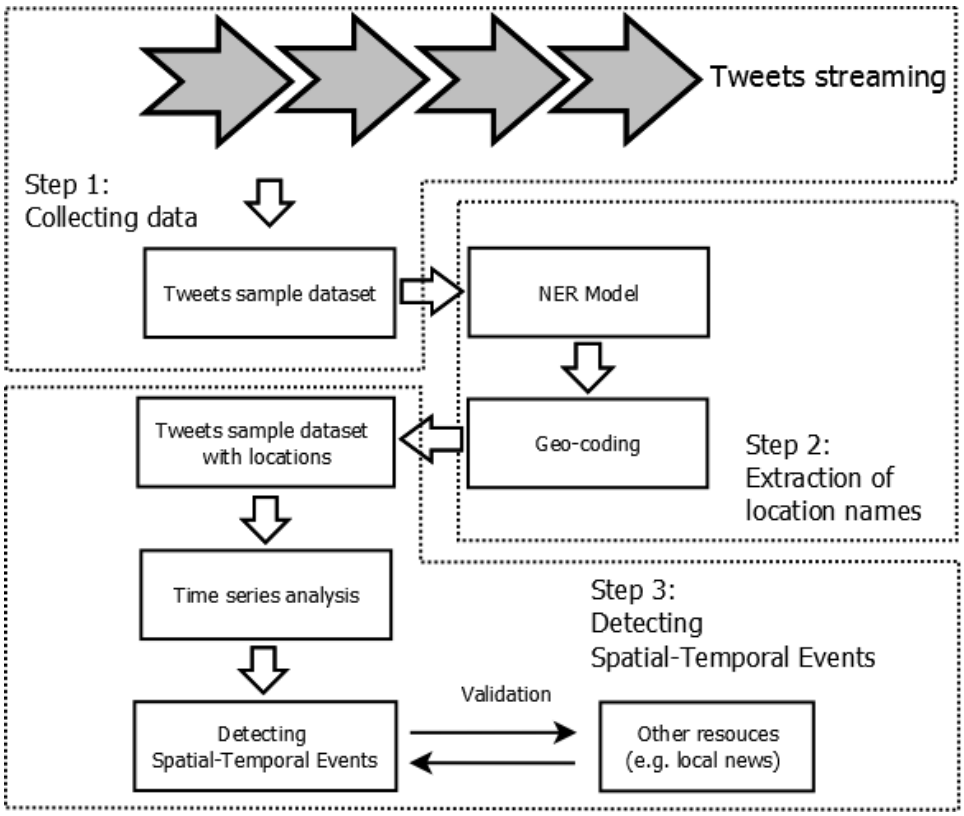


Figure 1 General model of detecting Spatial-Temporal Events via NER and time-series analysis

This model could be divided into three steps: collecting data, extracting locations and detecting Spatial-Temporal Events. In following paragraph, the details of each part will be described in more detail.

3.1.1 Collecting Data

The only data resource of this thesis was the tweets data. For the further analysis, the tweet data we needed should follow three rules:

- Continuity
  First, for detecting Spatial-Temporal event, especially for the temporal aspect, the collecting of tweets should be continued. Second, since the time difference among the population who post tweets in English, to collect all the information from them, the

collecting of tweets should be 24 hours per day as well, not USA time based or UK time based.

- Sample size
  Since the tweets data is very big, which beyond the ability of personal computer, it is impossible to analyse all tweets in this research. We had to make a proper sample. The sample size should be big enough to guarantee the randomicity and small enough for fitting the capability of personal computer.
- Language
  Twitter is a world-wide social networking platform that contains tweets in all main languages. The tweets data to be collected should be only in English. First reason is that it is the most understandable language for researcher; second is that for the NER analysis, the NER model on English is far more mature than it on other languages.

The collection of Tweets dataset began from the Twitter website. Twitter allows users and developers to access the dataset of tweets by Twitter Application Programming Interface (Twitter API) [1]. The Twitter API offers several interfaces to achieve tweet data for different purposes, one of these interfaces, which is called streaming API, was used in this thesis. The Streaming API, as part of Twitter API, provides a stream of tweets data available for all developers. To access these data, the streaming API will build a connection to the server of Twitter, then we could process these data at the programming environment of Streaming API. The Streaming API works in the way shown in Figure 2:

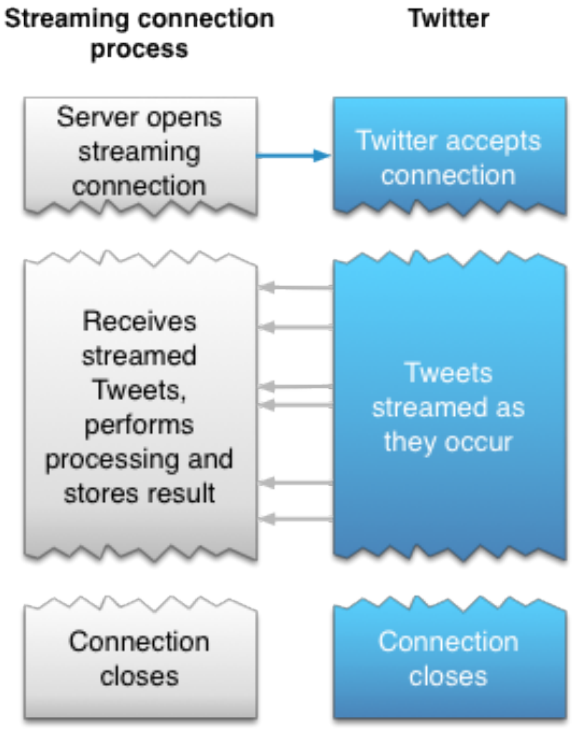


Figure 2 How Streaming API achieve twitter data[2]

These connections between Streaming API and Twitter could last a continuing period long enough to collect a large dataset, which fit the requirement of continuance. At the Streaming API side, we could do many processes on the tweets we got, such as filtering by language and

[1] https://dev.twitter.com/
[2] Figure was got from https://dev.twitter.com/docs/streaming-apis

selecting data randomly, these functions could fit the requirements about language and randomness.

After the establishment of these connections, we could store tweets in a dataset at the Streaming API side. As an interface which cannot run itself, the streaming API needs a programming environment to rely on. In terms of Python as an example, the data collection part was shown as Figure 3:

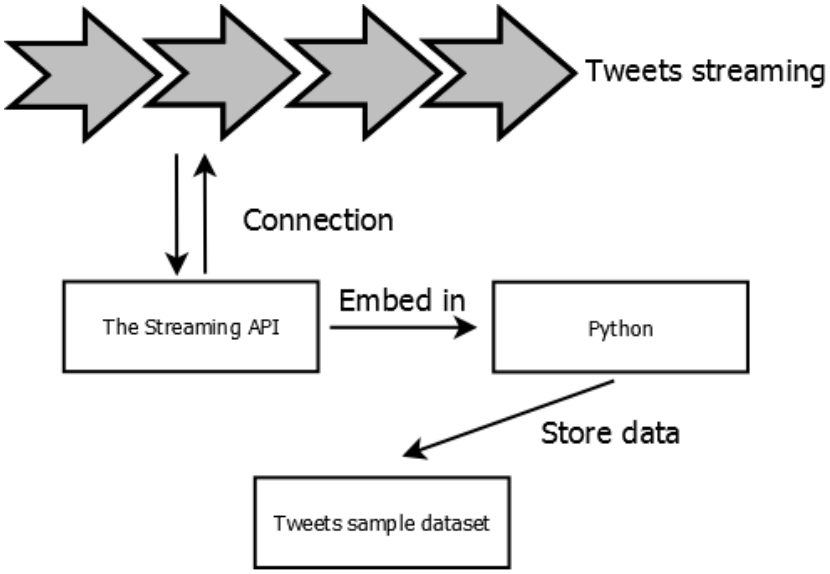


Figure 3 Collecting Tweets from Twitter

3.1.2 Extracting Location Names

One of the highlighted methods of this research was to extract locations from tweets with NER model, which aims on analysing properties of human's natural language and figure out possible words/phrases from sentence as names of location. The output of the NER model were all the location names extracted from our tweets dataset, and each location name was attached to the tweet which it came from. The detailed theory of NER model will be introduced in the next chapter.

However, the location names, even though we could recognise where they are, we needed give them coordinates so that they could be geo-analysed. Geo-coding service was applied in this research, to convert the input location name to the output coordinates.

The geo-coding service, not only provides coordinates, but also provides a structured address of this location name, for instance the city, province/state and country name of this location. This information was used to hierarchize these locations. In this thesis, finally the hierarchizing of location names was set as the primary method instead of coordinates, more detail are available in chapter 3.2.2.

3.1.3 Detecting Spatial-Temporal Events

After collecting the tweets and the location names, Spatial-Temporal Events could be extracted from these tweets. Both the spatial factor and the temporal factor would be analysed in this step. For the spatial factor, the tweets were classified based on different location names, then for temporal factor of one location, a time-series analysis was used to extract properties of these tweets dataset with the change on time axis.

The time-series analysing method used in this step contained two steps as shown in Figure 4: time-series decomposition and seasonal adjustment. The time-series decomposition divided the tweets data into three parts: seasonal, trending and remainder components, then the seasonal component was removed during the seasonal adjustment. The rest components, include trending and remainder components, showed the non-seasonal trends of tweets data, where we believed that Spatial-Temporal Evens were hidden in.

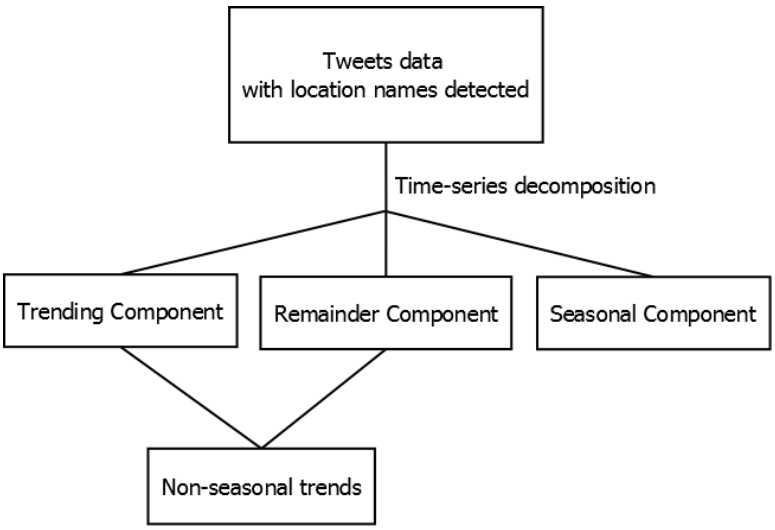


Figure 4 Time-series decomposition and seasonal adjustment

The non-seasonal trends of tweets, which was the result of time series analysis, then was manually analysed to detect whether there was a Spatial-Temporal Events hided inside. To validate these Spatial-Temporal Events, we checked the news online related to that location, at that time, with that topic.

3.2 THEORETICAL DESIGN

3.2.1 Basic Definition

To make the definition of Spatial-Temporal Events clear, we need to be aware of the difference between the term *event* and *event occurrence*. The term *event occurrence* means the happening of an event on an instant of time, while the *event* could be defined as a set of *event occurrence*(Zhang & Unger, 1996). For example, the "eat an apple" is an event that lasts a few minutes, where a few minutes means a period of time, in other words, a set of points on timeline; the *event occurrence* of eating an apple is "a bite on an apple with the mouth" which is only a point on timeline.

We define the occurrence of a Spatial-Temporal Event ($EventOccurence_{st}$) as a tuple of a spatial factor, a temporal factor and the content of this event,

$$EventOccurence_{st} = < T_h, L_e, E_t >$$

Where T_h is the temporal factor, meaning the happening time instant of this event, L_e is the location where this event happens, E_t is an event identifier which tells the content of this event occurrence (earthquake, traffic jam, fire or et al) and distinguishes this event occurrence from the others happened on the same time and location.

A tweet from Twitter streaming was originally endowed with temporal factor since it has the posting time as an attribute. We only needed to consider the spatial factor and the event identifier. To simplify the detecting of Spatial-Temporal Events from tweets, the following assumptions were made:

Assumption 1: Each tweet represents a temporal event occurrence.

This means the temporal factor and event identifier always exist in every tweet. While the temporal factor is naturally gained, the existing of event identifier means the tweet always tells a happening of something.

Assumption 2: Each tweet which could be extracted one location name represent one Spatial-Temporal Event occurrence.

This means the location name in a tweet is exactly where the event occurrence happens. In a given tweet, L_e, once it exists, always belongs to the $EventOccurence_{st}$ whose T_h and E_t were found from this tweet.

Assumption 3: Each tweet with multiple location names respectively represent multiple Spatial-Temporal Event occurrences.

This means for a given tweet, multiple L_e represent a set of different $EventOccurence_{st}$ with different L_e but shared the same T_h and E_t, these $EventOccurence_{st}$ are independent to each other. We ignored the hidden relations among the multiple location names mentioned in the same tweet. The complex $EventOccurence_{st}$ which happened at different locations was simplified to multiple simple $EventOccurence_{st}$ in this research.

Based on these assumptions, since the *event* is a set of qualified *event occurrence*, the Spatial-Temporal Event are able to be described as a set of qualified Spatial-Temporal Event occurrences. Here the qualifications were represented as an event identifier, a threshold of spatial factor and a threshold of temporal factor. The Spatial-Temporal Event ($Event_{st}$) could be defined as,

$$Event_{st} = \{EventOccurence_{st} | EventOccurence_{st} \in < T_{threshold}, L_{threshold}, E_0 >\}$$

Where the $T_{threshold}$ means the temporal interval which is a set of time instants, it describes a predefined length on timeline. The $L_{threshold}$ means an area, of a set of areas where the event happens, for example, for the event "square concert", the threshold $L_{threshold}$ is the area of that square where the concert happened; In terms of the event "All national museums are free for public today", the $L_{threshold}$ is a set of areas consist of the area of every museum in this country. Once there are different events happening at the same time within the same area, E_0 as an identifier, prevent this event being confused with others.

3.2.2 Spatial Factor

3.2.2.1 Extracting Spatial Factor by NER

In general, most of the research extract the spatial factor: position from tweets via geo-targeting. As we discussed in the introduction chapter, there were a few problems that may occur when using the Geo-targeting. As we introduced in the literature review, The NER was used to extract location names.

Since this thesis did not focus on the development of a NER model itself, a pre-designed toolkit or program of NER was needed. Examples of existing NER models include the Illinois Named Entity Tagger (Ratinov & Roth, 2009), the Natural Language Toolkit (Bird, Klein, & Loper, 2009) and the Stanford Named Entity Recognizer (Finkel, Grenager, & Manning, 2005). The Stanford Named Entity Recognizer (Stanford) NER was chosen in this thesis for its advanced modelling, abundant training sets and supporting documents.

The Stanford NER is a statistical model in the domain of semantic analysis, which aims to classify the words/phrases in natural language into different types. The natural language, in this case, are the unpremeditated language used, influenced and developed by human

naturally. It is distinguished from the constructed language or formal language, which were created by human unnaturally and purposely. The Table 1 showed an example of these three languages. It is obvious that most of the tweets are written by natural language,

Type of Language	Example
Natural	"Hello, how are you?"
Constructed	"Kiel vi fartas?" [1]
Formal	$e^{i\pi} + 1 = 0$

Table 1 Example of different languages

The Stanford NER provides a ready-made program, code resources and pre-trained datasets for named entity recognition. It is an implementation of linear chain Conditional Random Field (CRF) sequence models proposed by Lafferty and et al (Lafferty, McCallum, & Pereira, 2001).

In general, the CRF sequence model is a discriminative undirected probabilistic graphical model, which is a *random field* conditioned on observations. Consider a random variable X ranged over data (word/phrase/sentences in Stanford NER) sequences to be labelled, and a random variable $Y = (Y_1, Y_2, Y_3, Y_4, \cdots, Y_n)$ which will only range over the sequences of labels, for example, the type of names to be targeted on word/phrase. The linear chain CRF sequence model was defined as an undirected graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ while V means a set of vertices (dots) and E means a set of edges in this graph:

Definition: Graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$, Y is indexed by the vertices of G, $Y = (Y_v)_{v \in V}$. Y_v obey the *Markov property* when conditioned on X, that is $\mathrm{P}(Y_v | X, Y_w, w \neq v) = \mathrm{P}(Y_v | X, Y_w, w \sim v)$ where $w \sim v$ means w and v are neighbours in G.

This definition defined a random field conditioned on X, even though the shape of graph G is not fixed, Lafferty and et al assumed the graph G was fixed as a line chain shown on Figure 5,

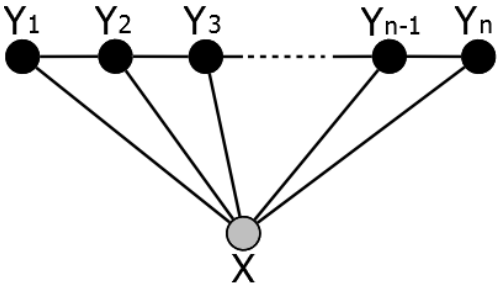


Figure 5 an example of linear chain CRF model

And for a given data sequence x, the probability of a label sequence y is:

$$P(y|x) \propto exp\left(\sum_{e \in E,k} \lambda_k f_k(e, y|_e, x) + \sum_{v \in V,k} \mu_k g_k(e, y|_v, y)\right)$$

Where $f_k(e, y|_e, x)$ and $g_k(e, y|_v, y)$ are two *feature functions* of the vertices and edges in the graph G, which need to be given and fixed before training, λ_k and g_k are the model parameters resulted after training.

[1] Esperanto, the most famous constructed language developed by L. L. Zamenhof.

The details of the training process of CRF sequence model beyond the purpose of this thesis report, but these process are fully realised in the Stanford NER, which provided a few CRF sequence models trained from some named entity recognition training dataset. With the help of these models, we could classify words and phrases in tweets into different names: person name, location name, organisation name and et al, then extract locations as the spatial factor of tweets.

3.2.2.2 Hierarchizing Spatial Factor

The spatial factor, in this case, is the location names extracted by NER. The raw location names contains two problems which impede their quality. First is the different spellings presenting the same location, including the informal or wrong spellings, for example, America, United States, US, U.S. and america refer to the same location, yet in different location names. Although it is not very hard for human to recognise these names as one location, it could be a big, even fatal problem for programs to process them.

The second problem is the unknown relations among different names. A full address contains a group of hierarchized names: road/street name, city name, province/state name, country/nation name. Our dataset of extracted location names included names at different levels, but we did not know which levels they are, and they were not linked to each other. For instance, we have city name “London” and nation name “UK”, but there is no link existing in our dataset to describe London is part of UK, when we are analysing the tweets about “UK”, the tweets with location name “London” should be included as well.

To solve these two problems, Geo-coding is necessary to be applied. The main function of Geo-coding is to convert human-readable locations into computer-readable coordinates. The big location name dataset provided by Geo-coding service, including spelling mistakes recovery function, could solve the first problem. Different names of one location will be converted to only one geographic coordinates.

In theory, the coordinates could also solve the second problem, but a simpler method could be used in this research, is an additional function of Geo-coding: Administrative level of one location. The Google Geo-coding service provides three administrative levels of one location: City, state and country. For a given location name to be Geo-coded, the Google Geo-coding service provides a long formal name and a short abbreviation name of the administrative levels above current location, and for the administrative level lower than current location, it will result empty. Using this method, our location names are hierarchized, three new fields are added to every location name: its city, state and country name. For analysing on a city level, we could easily apply a query that select all location names whose city name is this city.

In summary, the Geo-coding was used to solve the two problems we mentioned, its spelling mistakes recovery function was used to solve the first one and its administrative level was used to solve second. The coordinates received from Geo-coding was not be used.

3.2.2.3 Presenting Tweets Dataset

The event identifier of a given tweet represents the content of the event that this tweet belongs to. Two tweets may have the same event identifier but phrased in different sentences. Because of the limited and knowledge, recognising the content of events from tweets were finished manually.

Since this work had to be done manually instead of automatically, to reduce the workload of manual work, the obtaining of event identifier should be done after all the filtering operated by computer. The temporal factor and spatial factor were extracted firstly to reduce the size of tweets data, then we needed to recognise some pattern of interests from the collection of

tweets with temporal and spatial factors. To our dataset, the pattern of interests, indeed, is an extraordinary change of amount of tweets within a short time, on a given location. It will highly reduce manual workload once we figure out the pattern of interests successfully.

To recognise pattern of interests for the analysis on next step, first we needed to present the data in a simple and strict way. Here we defined as a three-dimensional space,

Lemma: Define a three-dimensional space $S^3 = (N, T, L)$, where N and T are continuous variables representing the amount and posting time of tweets, L is a set of location names extracted from tweets. Then all the tweets with location(s) can be mapped in this space.

In the practical application, the L axis in the space S^3, as a set of extracted location names, is a discrete and finite set. For a given value of L_0, the three-dimensional space S^3 could be reduced to a two-dimensional $S^2 = (N, T)$. Since we ignored the complex spatial-temporal events with different locations at assumption 3, the links between different two-dimensional spaces S^2 are not existed. All the spatial-temporal event occurrences detected from tweets (tweets with locations) are exist in a set of two-dimensional spaces:

Theorem: Define a set $R_{st} = \{(N_i, S_i, L_i) | i = 1,2,3, \cdots, m\}$ where m is the total amount of location names extracted from tweets. Based on the assumption 3, all the tweets with locations could be mapped on R_{st}.

Figure 6 is an example diagram of set R_{st},

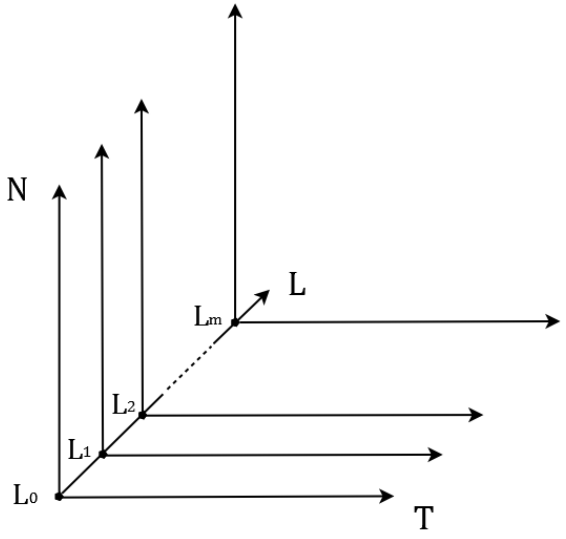


Figure 6 an example diagram of set R_{st}

Now we consider a two-dimensional space $S^2 = (N, T)$ for a given L_0, which described the change in amount of tweets through time, corresponding to L_0. These data point on S^2 generated a time series, to detect the pattern of interests from them.

3.2.2.4 Time-series Decomposition and Seasonal Adjustment

The tweeting behaviour of public contains two parts: the regular tweets that are posted every day and the irregular tweets that are posted for special reasons, issues or events. To classify tweets dataset into these two parts at the quantity level, we needed to remove the number of tweets appeared seasonally from the original data. A time series decomposition was applied on our dataset. The purpose of time series decomposition is to divide a time series into a seasonal, a trend and an irregular component. Suppose a time series contains N values, for i =

1 to N, the time-series decomposition can be described as dividing the original data into a linear combination of three components:

$$Y_i = T_i + S_i + R_i$$

Where Y_i means the value in time series; T_i means the trending component which describes a long-term tendency of the time series; S_i means the seasonal component, which tells a seasonal pattern repeating per period (day, month, quarter or year) and R_i means remainder, which is the remnant out of the seasonal value and general trend value. Seasonal adjustment means to remove the seasonal component out of this combination, then the non-seasonal trends was given as,

$$NST_i = T_i + R_i$$

Where NST_i means the non-seasonal trends.The key point of this step, of course, is to decompose the time-series. There are several time-series decomposition models available. In this case, we used a Seasonal-Trend decomposition procedure based on Loess (STL), which aims to decompose time-series into a liner combination of seasonal, trend and remainder components using locally-weighted regression (Loess) method (Cleveland, Cleveland, McRae, & Terpenning, 1990). Comparing to other models, the advantages of STL model, which drove us to make this choice, are its robustness on aberrant values and its feasibility on any type of seasonality.

The STL is based on locally-weighted regression (Loees), this regression could be briefly described as a loess regression curve $\hat{g}(x)$ for n sets of independent and dependent measurement variables: x_i and y_i , $i = 1\ to\ n$. The calculation of $\hat{g}(x)$ is just like this,

First we define a *tricube weight function* W:

$$W(u) = \begin{cases} (1-u^3)^3 & for\ 0 \le u\ < 1 \\ 0 & for\ u \ge 1 \end{cases}$$

Then we give a positive integer q. The *neighbourhood* is defined as the q values of x_i which are closest to x. When $\le n$, let $\lambda_q(x)$ be the q th farthest distance from x_i to x. The *neighbourhood weight* $v_i(x)$ of each x_i could be given based on the distance from x_i to x and the farthest distance $\lambda_q(x)$,

$$v_i(x) = W\left(\frac{|x_i - x|}{\lambda_q(x)}\right)$$

The $v_i(x)$ get larger value when the x_i is closing to x and become zero when x_i is at the qth farthest point. If $q > n$, let $\lambda_n(x)$ be the distance from farthest x_i to x, then we define $\lambda_q(x)$ for $q > n$ as,

$$\lambda_q(x) = \lambda_n(x)\frac{q}{n}$$

Then we give a *degree* d which imply the degree of the polynomial for fitting the observation data with the *neighbourhood weight*, for instance, $d = 1$ means locally-linear fitting and $d = 2$ means locally-quadratic fitting, The value of this fitted polynomial at x is $\hat{g}(x)$.

It is obvious that $\hat{g}(x)$ tends to smoother when the neighbourhood q becomes larger. When q is tending to infinity, the $v_i(x)$ will tend to 1 and the $\hat{g}(x)$ will tend to the ordinary least-square polynomial fits with degree d.

The STL time-series decomposition model contains two recursive procedures: the inner loop and outer loop. Let's go back to the previous definition of time series decomposition,

$$Y_i = T_i + S_i + R_i$$

Inner loop: Let T_i^k and S_i^k be the seasonal and trending component after kth pass of inner loop, For the $(k+1)$th pass of inner loop[1],

Step 1: Detrending

Trending component is removed from time-series to calculate the detrended series: $Y_i - T_i^k$. For the first pass, the T_i^o is initialised zero.

Step 2: Cycle-subseries smoothing

Every cycle-subseries of detrended series is loess smoothed with $q = n_s$ and $d = 1$, where n_s means the smoothing parameter for seasonal component. Here the cycle-subseries means the series consist of the same time position repeatedly appears in every cycle, for example, a time-series with monthly observation data from 1990 to 2000, for n_p, which is the number of observations in each cycle of the seasonal component, is 12, the January subseries means all the data of January from January 1990 to January 2000. Before the loess smoothing on cycle-subseries, each cycle-subseries is extended with one time position before the first one and one position after the last one, for the example just mentioned, the loess smoothing is applied on position of January from January 1989 to January 2001.

Let $C_i^{(k+1)}$ be a collection of all smoothed cycle-subseries where i = 1 to N, then $C_i^{(k+1)}$ is a temporal seasonal series that contains all smoothed values at positions ranged from $(-n_p + 1)$ to $(N + n_p)$, totally $N + 2n_p$ values.

Step 3: Low-pass filtering of smoothed cycle-subseries

A low-pass filter is applied on the smoothed cycle-subseries $C_i^{(k+1)}$, this filter contains (must be in this order),

1. Two times moving average with length n_p;
2. One time moving average with length 3;
3. One time loess smoothing with $q = n_l$ and $d = 1$

Here n_l means the smoothing parameter for the low-pass filter, let $L_i^{(k+1)}$ be the output of this filter. Since the moving average could not reach the end of series, in $L_i^{(k+1)}$, the i is defined from 1 to N, this is the reason that we extends $2n_p$ values in step 2.

Step 4: Detrending of smoothed cycle-subseries

The seasonal component of $(k+1)$th pass is $S_i^{(k+1)} = C_i^{(k+1)} - L_i^{(k+1)}$, the low-frequency power was blocked from entering seasonal component.

Step 5: Deseasonalizing

A deseasonalized series was calculated by $Y_i - S_i^{(k+1)}$.

Step 6: Trend smoothing

[1] One important feature of STL is its capability on processing data with missed values, since the dataset of this thesis was completed, the methods to process missed values were ignored in the description of inner/outer loops.

The deseasonalized series resulted from step 5 then is loess smoothed with $q = n_t$ and $d = 1$, where n_t is the smoothing parameter for the trending component. Trending component of this pass: T_i^{k+1} is set to the result of this loess smoothing and enter the next pass.

The seasonal and trending component are updated in these six steps of inner loop.

Outer loop: In terms of outer loop, the robustness of weights are calculated to reduce the transient or aberrant behaviour on seasonal and trending component, in other words, the outer loop drives more non-robust values enter remainder component from the trend/seasonal components. First we define the remainder component: After the running of the first pass of inner loop, the remainder component is,

$$R_i = Y_i - S_i - T_i$$

Then we need to define a threshold to find the outlier, let $l = 6\,median(|R_i|)$, the remainder larger than l will be considered as outlier, so the time position that cause this outlier should get smaller weight, here we define a *bisquare weight function* BW,

$$BW(u) = \begin{cases} (1-u^2)^2 & for\ 0 \le u\ < 1 \\ 0 & for\ u \ge 1 \end{cases}$$

Then we give the *robustness weight* as,

$$\mu(i) = BW\left(\frac{|R_i|}{l}\right)$$

The outer loop will repeat all the steps of inner loop, yet at step 2 and step 6, the neighbourhood weights used in loess smoothing need to be multiplied by the robustness weight at that time position.

Besides the number of passes through these two loops, six parameters shown in Table 2 need to be decided for STL decomposition model:

Model Parameters	Description
$\boldsymbol{n_p}$	The number of observations in each cycle of the seasonal component
$\boldsymbol{n_i}$	The number of passes through the inner loop
$\boldsymbol{n_o}$	The number of robustness iterations of the outer loop
$\boldsymbol{n_l}$	The smoothing parameter for the low-pass filter
$\boldsymbol{n_t}$	The smoothing parameter for the trending component
$\boldsymbol{n_s}$	The smoothing parameter for the seasonal component

Table 2 Model parameters of STL model

In this thesis, our data source is the hourly amount of tweets related to one location names, a proper value of $\boldsymbol{n_p}$ **is 24** which implies that 24 hour (one day) will be treated as one season in extracting seasonal component. It is not hard to forecast that people's tweeting behaviour is daily repeating: More tweets on leisure time and less tweets on deep night. The result of seasonal component in Figure 9 shows that one day is a proper season for tweets time-series analysis.

Three recommendations of the values of $\boldsymbol{n_i}$ and $\boldsymbol{n_o}$ were given by designers of this STL model, which are $\boldsymbol{n_i = 2, n_o = 0}$, $\boldsymbol{n_i = 1, n_o = 5}$ and $\boldsymbol{n_i = 1, n_o = 10}$. The spatial-

temporary events, especially the event which happens suddenly, would cause a wide discussion via a large amount of Tweets. The number of these tweets will beyond the seasonal components and long-term trend, it means most of these tweets will be classified as remainder component. To make the STL model more sensible for the remainder component, the robustness iterations were necessary, so we needed to give $\boldsymbol{n_o}$ a value other than zero. For $\boldsymbol{n_i = 1}$, the designers of STL model found that $\boldsymbol{n_o = 5}$ is safe enough for the convergence, and near certainty of convergence would be provided by the setting $\boldsymbol{n_o = 10}$. For no evidence showed the requirement of $\boldsymbol{n_o = 10}$ is necessary in our tweets datasets, so we used the setting $\boldsymbol{n_i = 1, n_o = 5}$.

The $\boldsymbol{n_l}$ was set the least odd integer greater than/ equal to $\boldsymbol{n_p}$, so it was $\boldsymbol{n_l = 25}$ in this case. The $\boldsymbol{n_s}$ should be an odd integer at least $\boldsymbol{7}$, the seasonal component will become smoother with the increasing of $\boldsymbol{n_s}$. In this thesis, it was set to 35.

The $\boldsymbol{n_t}$ determines the degree of smoothing on trending component, it should be set as the least odd integer satisfying this inequality,

$$n_t \geq \frac{1.5 n_p}{1 - 1.5 n_s^{-1}}$$

Since $\boldsymbol{n_p = 24}$ and $\boldsymbol{n_s = 35}$, in this thesis, $\boldsymbol{n_t}$ was set to $\boldsymbol{39}$.

The pattern of interests would be the part out of ordinary in the curves of seasonal component and remainder component. They would be detected by manually work with the assistances of basic mathematical operation (differential coefficient and et al).

3.3 IMPLEMENTATION

3.3.1 Collecting Data

The streaming API provides a low latency access to global public stream of tweet data, and returns the tweets in the data format JSON (JavaScript Object Notation).

The streaming API can be applied in most of the popular programming languages. In this thesis, the Python[1] was chosen to realize it. The python package for streaming API: tweepy[2] was used to process tweets data on python platform.

With the help of streaming API, the user of Twitter could not only use elementary functions such as post a tweet or retweet, but also use the advanced developing functions, for instance, accessing tweets by programming or filter tweets by keywords. To make the different users clearer, in following paragraphs, we use “user” referring the public normal users of Twitter, and use “developer” to refer to the user who use the programming methods, such as streaming API, to access and process tweets data.

The amount of tweets that one developer could access is limited: If there are too much requests by one developer within a short time, it would be considered as a malware, the authorization of this developer may be forbidden. To avoid the happening of this, we had to filter the tweets before we got them. There are several methods offered by tweepy to filter the tweets based on their keywords or other attributes. Any filtering on keywords was rejected in this thesis, for it might cause the deviation on the randomness of our tweets dataset. Finally, the “sample” method offered by streaming API was selected for filtering, which would

[1] http://www.python.org/
[2] https://github.com/tweepy/tweepy/

randomly select a small sample tweets (around 1% of total tweets on streaming) from the current global stream of tweets.

The random function returned the tweets by the JSON data structure. Here is an example of one tweet returned by Streaming API in JSON structure:


```
{"created_at":"Tue Feb 11 14:03:30 +0000
2014","id":433239876578717696,"id_str":"433239876578717696","text"
:"I will go back to Wageningen University
tomorrow.","source":"web","truncated":false,"in_reply_to_status_id
":null,"in_reply_to_status_id_str":null,"in_reply_to_user_id":null
,"in_reply_to_user_id_str":null,"in_reply_to_screen_name":null,"us
er":{"id":764728722,"id_str":"764728722","name":"\u5218\u94ba","sc
reen_name":"liuyue1989","location":"","url":null,"description":nul
l,"protected":false,"followers_count":3,"friends_count":13,"listed
_count":0,"created_at":"Sat Aug 18 00:33:33 +0000
2012","favourites_count":0,"utc_offset":0,"time_zone":"London","ge
o_enabled":true,"verified":false,"statuses_count":8,"lang":"en-
gb","contributors_enabled":false,"is_translator":false,"is_transla
tion_enabled":false,"profile_background_color":"C0DEED","profile_b
ackground_image_url":"http:\/\/abs.twimg.com\/images\/themes\/them
e1\/bg.png","profile_background_image_url_https":"https:\/\/abs.tw
img.com\/images\/themes\/theme1\/bg.png","profile_background_tile"
:false,"profile_image_url":"http:\/\/pbs.twimg.com\/profile_images
\/2516088362\/image_normal.jpg","profile_image_url_https":"https:\
/\/pbs.twimg.com\/profile_images\/2516088362\/image_normal.jpg","p
rofile_link_color":"0084B4","profile_sidebar_border_color":"C0DEED
","profile_sidebar_fill_color":"DDEEF6","profile_text_color":"3333
33","profile_use_background_image":true,"default_profile":true,"de
fault_profile_image":false,"following":null,"follow_request_sent":
null,"notifications":null},"geo":null,"coordinates":null,"place":n
ull,"contributors":null,"retweet_count":0,"favorite_count":0,"enti
ties":{"hashtags":[],"symbols":[],"urls":[],"user_mentions":[]},"f
avorited":false,"retweeted":false,"filter_level":"medium","lang":"
en"}
```


To fit the different purposes of users, Twitter streaming API offers almost all available properties of the tweet and the user who posted it. In terms of this research, we only needed three properties of one tweet (coloured in red): the creating time (`"created_at"`), textual content (`"text"`) and the language (`"lang"`). You may notice that there are more than one properties named `"created_at"` in this example, this is because it contains the information about both the tweet and the user. The first `"created_at"` is the first level property belongs to the tweet directly, while the second `"created_at"` is the sub-property of the first level property `"user"`. The `"text"` and `"lang"` are similar to `"created_at"`. The three properties that we kept referred to the properties of tweet, not user.

The `"lang"` was used to filter out all tweets which were not in English. To use the storage space more economically, after language filtering, we removed all the properties other than creating time and textual content during saving data.

JSON is a new and good structure for data exchange, yet it is not very easy to read and understand by people's traditional prospective. To make the viewing friendlier to human, we used the standard Microsoft Excel data structure: XLS to store tweets data. Because of the limit of rows of an XLS file, we stored 30000 tweets into one XLS file, then generated a new

XLS file and named it automatically by programming. The Python package “xlrd” and “xlwt” were used to create and process XLS file in Python[1].

Finally, at this step, a small Python program was designed: it could keep accessing a small sample of current tweets with low latency by the help of tweepy and Twitter Streaming API, then all the non-English tweets were removed, at last, we stored the useful attributes of these tweets (created time and textual content) into XLS files. The code of this program was shown in appendix A.

During the observation season, we kept this program running without pausing to collect tweets data. The output XLS files were considered as the raw data resource for future steps, Table 3 shows an example of our raw tweets dataset,

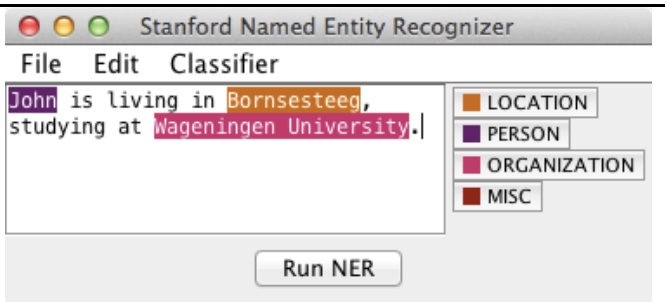
Time	Text
Tue Nov 05 09:07:40 +0000 2013	RT @kiansmahone: #voteaustinmahone got a beach house I could sell you in idaho

Table 3 An example of one tweet in dataset.

As we can see from this table, the property “time” was stored in a standard Python time structure for next steps. Since we explored the tweets in the language English, only basic letters, numbers and punctuation symbols were required, so that the text of tweet will be transformed and stored in the commonest character-encoding scheme: ASCII (American Standard Code for Information Interchange).

3.3.2 NER analysis

After achieving the tweets data, the NER analysis was applied on the text of tweet to extract the latent location information inside, the Stanford NER was selected for NER. The Stanford NER is a well-developed working environment develop by Java, which contains both the Graphic User Interface (GUI) and programming codes supporting. The output of Stanford NER was provided in several formats, the Table 4 showed four different output formats of Stanford NER.

Coloured GUI Output	
Output formatted by Slash tag	John/PERSON is/O living/O in/O Bornsesteeg/LOCATION ,/O studying/O at/O Wageningen/ORGANIZATION University/ORGANIZATION

[1] http://www.python-excel.org/

Output formatted by inlineXML	<PERSON>John</PERSON> is living in <LOCATION>Bornsesteeg</LOCATION>, studying at <ORGANIZATION>Wageningen University</ORGANIZATION>
Output formatted by xml	<wi num="0" entity="PERSON">John</wi> <wi num="1" entity="O">is</wi> <wi num="2" entity="O">living</wi> <wi num="3" entity="O">in</wi> <wi num="4" entity="LOCATION">Bornsesteeg</wi><wi num="5" entity="O">,</wi> <wi num="6" entity="O">studying</wi> <wi num="7" entity="O">at</wi> <wi num="8" entity="ORGANIZATION">Wageningen</wi> <wi num="9" entity="ORGANIZATION">University</wi>

Table 4 Four Different output formats of Stanford NER

We could read from the table that the first output format: Coloured GUI format was designed for understanding of human, while the other three formats were designed for information exchange by programming. We used the inlineXML as the outputting format, then automatically deal it with programming.

Before inputting our tweets dataset to the Stanford NER, we need to choose a classifier. A classifier is a set of parameters of NER model which is result of training on training dataset. In this thesis, we used a classifier with 3 class: Location, Person and Organization. This classifier was provided by Stanford NER, it was designed to be a combination of two models trained by different datasets: one is a 4 class model (Location, Person, Organization, Misc[1]) trained for CoNLL (Conference on Computational Natural Language Learning), another one is a 7 class model (Time, Location, Organization, Person, Money, Percent, Date) trained for MUC (Message Understanding Conference).

A Java program was written in this step to input tweets based on the java package offered by the Stanford NER. This program automatically visited all the XLS files which stored all the tweets we collected during the data preparing. Then this program applied the Stanford NER model and 3 class classifier to found out the location names within the text of every tweet. The code of this Java program was available on appendix B.

Since the purpose of this research is about tweets with location information, for the tweets with no location name detected by Stanford NER model, they were removed from our datasets after this step. The remaining tweets were saved in XLS file, an example of data structure was shown in appendix C.

3.3.3 Geocoding

The location names, such as “London”, “New York” or “Amsterdam Central Station”, were extracted from tweets by Stanford NER model. However, they were still not geo-data, since we did not give coordinates yet. In this step, the geocoding service was used to translate these location names into coordinates. There are a lot of geocoding services available, some of them are offered by the internet mapping service, such as Google Map and Bing Map, because it is necessary for internet mapping operator to response to the address typed by users, and link

[1] Miscellaneous, Misc for abbreviation.

these address to right coordinates on map. In this thesis, the Google Geocoding API[1] was selected as the geocoding method.

The pygeocoder[2], which is a plug-in package on Python, was used to access the Google Geocoding API. A Python program was developed in this step. First, this Python program collected all the unique location names from the result of NER analysis; Second, it visited the Google Geocoding API with the help of pygeocoder to process these location names; At last, the result was stored into a dataset as an XLS file, and the location name was the unique key of this dataset.

The Google Geocoding API returned lots of information related to one location name, for example, here is a result of Google Geocoding API of the location name "Wageningen":

```
[{u'geometry': {u'location_type': u'APPROXIMATE', u'bounds':
{u'northeast': {u'lat': 52.0007417, u'lng': 5.7243482},
u'southwest': {u'lat': 51.9363499, u'lng': 5.6058803}},
u'viewport': {u'northeast': {u'lat': 52.0007417, u'lng':
5.7243482}, u'southwest': {u'lat': 51.9363499, u'lng': 5.6058803}},
u'location': {u'lat': 51.9691868, u'lng': 5.6653948}},
u'address_components': [{u'long_name': u'Wageningen', u'types':
[u'locality', u'political'], u'short_name': u'Wageningen'},
{u'long_name': u'Wageningen', u'types':
[u'administrative_area_level_2', u'political'], u'short_name':
u'Wageningen'}, {u'long_name': u'Gelderland', u'types':
[u'administrative_area_level_1', u'political'], u'short_name':
u'GE'}, {u'long_name': u'The Netherlands', u'types': [u'country',
u'political'], u'short_name': u'NL'}], u'formatted_address':
u'Wageningen, The Netherlands', u'types': [u'locality',
u'political']}]
```

Not only the coordinates, but also the administrative area levels (city, province and country) were useful for our research, so in these step, they were all stored. It is useful to keep city, province and country names because it is more convenient for further statistics. For instance, if we want to detect the events in Netherlands, the location names we need to take care should contains both the names directly referring to Netherlands (Netherlands, Holland, Nederlands) and the names of location in Netherlands (Dutch address, city or province name). Keeping the administrative area levels aimed to make the relations between different location names clearer. Table 5 shows the final structure that the geocoding result was stored and:

Location Name	Latitude	Longitude	city	city_short	province	province_short	country	country_short
Amsterdam	52.37022	4.895168	Amsterdam	Amsterdam	North Holland	NH	The Netherlands	NL

Table 5 One row of location names after Geo-coding

3.3.4 Time-series Decomposition and Pattern of Interests

The time-series decomposition steps was realised based on the tweets data with geo-coded location names in R [3], which is a software/developing environment for statistical computing and graphics. For the spatial factor, as we introduced in the theoretical design, all our tweets data could be shown in a set of two-dimensional space. First of all, we divided our dataset to different classes based on the locations. As it was discussed in the definition of Spatial-

[1] https://developers.google.com/maps/documentation/geocoding/
[2] https://pypi.python.org/pypi/pygeocoder
[3] http://www.r-project.org/index.html

Temporal Event, the threshold of positions needed to be defined. For the size of our data is global, hardly could we see two adjacent locations existing in the dataset, so the threshold of positions was set at the country level first, then went into province level when it was necessary.

As we introduced before, the hierarchizing of location names was made by geo-coding processing, for the United Kingdom's example, we even did not need to define a threshold box of United Kingdom with coordinates, a selection with all the locations whose "country name" is United Kingdom was a perfect threshold of locations in the UK.

Then for a given location, the original temporal factor of tweets was in scale "sec". According to the size of our data, before we did the time-series decomposition, we changed this scale to "hour": Summing the amount of tweets in one hour.

Then we created a time-series object from our tweets data using the function: ts() of R package, while x-axis was the time sequence in the scale of hour, y-axis was the amount of tweets per hour. Then we input the tweets of one location into a STL time-series decomposition model: STL (time-series object)[1], the parameters were set as we introduced in the theoretical design part.

The seasonal adjustment was applied on the result of time-series decomposition to produce non-seasonal trends of tweets data, then we could extract pattern of interests from the data curve. This analysis was mainly finished manually with the help of simple calculations, such as *lagged difference* $(x_{n+lag} - x_n)$, for within a limited time, the human was better than computer to recognise strange/outlier part of a curve.

The R codes for this step were shown at appendix E.

3.3.5 Detecting Spatial-Temporal Events

In terms of the pattern of interests recognised from the time-series decomposition, the event identifier was extract by manually checking the patterns of interests. This checking work aimed on being aware of the event identifier, in another word, which topic makes the distribution of tweets different at that moment in time. It was finished like this: we analysed the event occurrences (tweets) with same spatial factor in a temporal factor threshold, then manually checking the texts of tweets to achieve the event identifier. The temporal factor threshold was set to one hour, which was one time unit in the time-series decomposition[2].

To validate these Spatial-Temporal Events, a search about related news on local Medias or websites was executed to check the event identifier, temporal factor and spatial factor.

[1] B.D. Ripley; Fortran code by Cleveland et al. (1990) from 'netlib'.

[2] More discussions about the selection of temporal factor threshold would be available on chapter 5.

CHAPTER 4: RESULT AND VALIDATION

4.1 DATA COLLECTING

From 09:07:40 of 5th November to 01:52:39 of 15th November, we collected 14,740,000 tweets via our data collecting program. Although we did not know how many tweets were posted in total in these days, using the prediction based on 500 million tweets per day, our sample dataset covered about 0.3% of all tweets during these eleven days. These tweets were saved in about 500 excel files having two attributes: Posting time and Text. They are sorted by the time order and all the texts are in English.

4.2 NER ANALYSIS

All the raw tweets data were sent into the NER model. Finally, 548,724 tweets were detected that contains location names, which covered 3.7% of our raw tweets dataset. 638,999 location names (containing duplicates) were extracted. After cleaning up these location names, 56,457 unique location names were extracted by the Stanford NER model. The top ten popular location names are shown in the Table 6:

Rank	Location names	Frequency
1	US	19499
2	London	18130
3	Philippines	15464
4	UK	12668
5	Argentina	11086
6	Austin	9214
7	America	8274
8	New York	6322
9	Texas	5553
10	U.S.	5484

Table 6 Top ten popular location names in NER result

In this step, even though we cleaned up duplicates caused by differences in letter case, such as uk, UK and Uk or America and america, more complex duplication such as US and U.S. was not cleaned. However, they would be cleaned at next step with the help of Geo-coding, since the administrative level given by Geo-coding is the best choice to clean such duplications.

Although we did not executed the validation of NER output, we recommended to do *hypothesis testing* on a sample of our NER results and raw tweets data. For a *null hypothesis*: location name exists in a tweet, an example of true positive (TP), true negative (TN), false positive (FP) and false negative (FN) are given in Table 7 based on the result of our Stanford NER model,

NER result (underlined words were the location names it extracted)	**Explains**
True Positive (TP)	
“RT @NatGeoID: Beautiful sunrise in Paris\n#TravelLovers #RomanticPlace http:\/\/t.co\/UahSFw7j9q”	Location name “Paris” was existed, correct and successfully extracted.
True Negative (TN)	

"LOL not good i wasted my three days"	There was no location name extracted, and accurately there was no location name in this tweet.
Type I error: False Positive (FP)	
"RT @jdbexistence: I like Austin, love his songs and the mahomies so yeah #voteaustinmahone"	The "Austin" was extracted, yet here the writer mean the singer Austin, not the city Austin in United States.
Type II error: False Negative (NP)	
"what a shame those elites are making for china"	The "china" at the end of this sentence was a location, but not extracted by the NER model, probably because of failing to type in the first letter of a country name in upper case.
"RT @4FreedominIran: demonstration in berlin against maliki presence in US #FreeThe7 #Iran @amnesty @dpa 62thday of hungerstrike http:\/\/t.c\u2026"	The "Iran" at the middle of this sentence was a location name, but failed to be extracted for the interfering from the hashtag # before it.

Table 7 Examples of TP, TN, FP and NP from the result of NER model[1]

There are a few reasons may cause Type I and Type II error, which will discussed more extensively in the discussion chapter. Because of the lack of human resource and limit of time, the further statistic on each type of error had not been done.

4.3 GEO-CODING

Top 4,000 location names were sent to Google Geo-coding service. There are two reasons that drive us to do not sent all location names to Geo-coding step: first is that Google Geo-coding service allows user to commit at max 2,500 requests per day; Second is that for most of our location names, they were not statistically meaningful because they only appeared a few times in the dataset, for instance, the 4,000th location name is "Kenora" (a small city in Canada), which appeared only 8 times.

165 Location names failed to be geo-coded out of 4,000 location names, the success rate was 95.88%. The errors caused by the Stanford NER model and Geo-coding service. For instance, The Stanford NER caused the errors by recognising a wrong name as location name, such as: "Timberland" (a brand name of outdoors wears, wrongly extracted by Stanford NER model), "Banana Republic" (a brand name of clothes, wrongly extracted by Stanford NER model); and the Geo-coding failed to recognise some new geographical concept, such as: "Greater New York City Area".

The administrative level was given to every location name by geo-coding, so that the US/U.S. were distinguished at this step. For example, at the national level, there were 330,595 tweets related to United States, which contains the different forms of "US" and all detected American location names, top three countries after geo-coding is shown in Table 8,

Country name	Frequency
United States	330595
United Kingdom	86714

[1] The tweets used in this table was in their original text, so spelling errors may occur.

Philippines	29689

Table 8 Top three names in country level

And an example of the Geo-coding result was shown in appendix D.

4.4 DETECTING SPATIAL-TEMPORAL EVENTS: CASE STUDY "UNITED STATES"

For detecting Spatial-Temporal Events, the United States was select as a case study.

4.4.1 Time-series Decomposition

All the tweets which were detected that contains the location names of/in U.S. were input in to STL timer-series decomposition model, the result was shown in Figure 7,

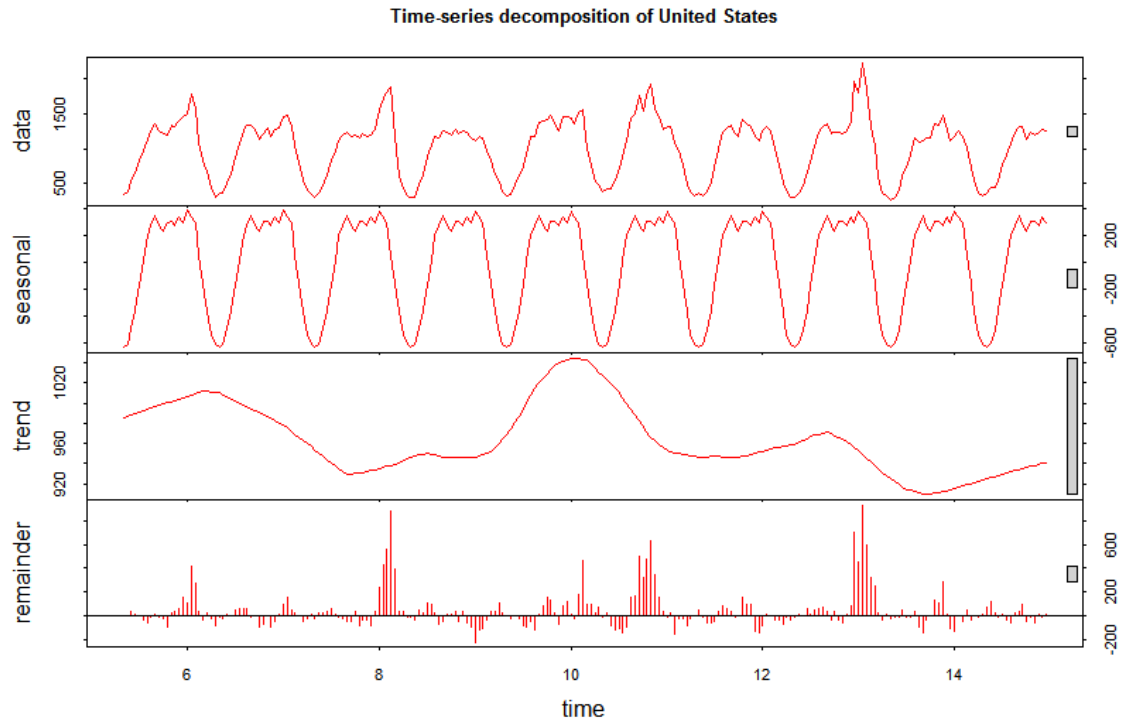


Figure 7 Time-series decomposition of United States from Nov. 5th to Nov. 15th, 2013

In this figure, the y-axis show tweets per hour in four components: data (input data), seasonal, trend and remainder. The x-axis is the time sequence in unit "day" ranged from 5th November to 15th November, while 24 observations exists in one unit of x-axis. The four grey rectangles at the right of figure showed the range of trending component at the scale of other three sub-figure.

The seasonal component was extracted by the time-series decomposition, it was shown in Figure 8,

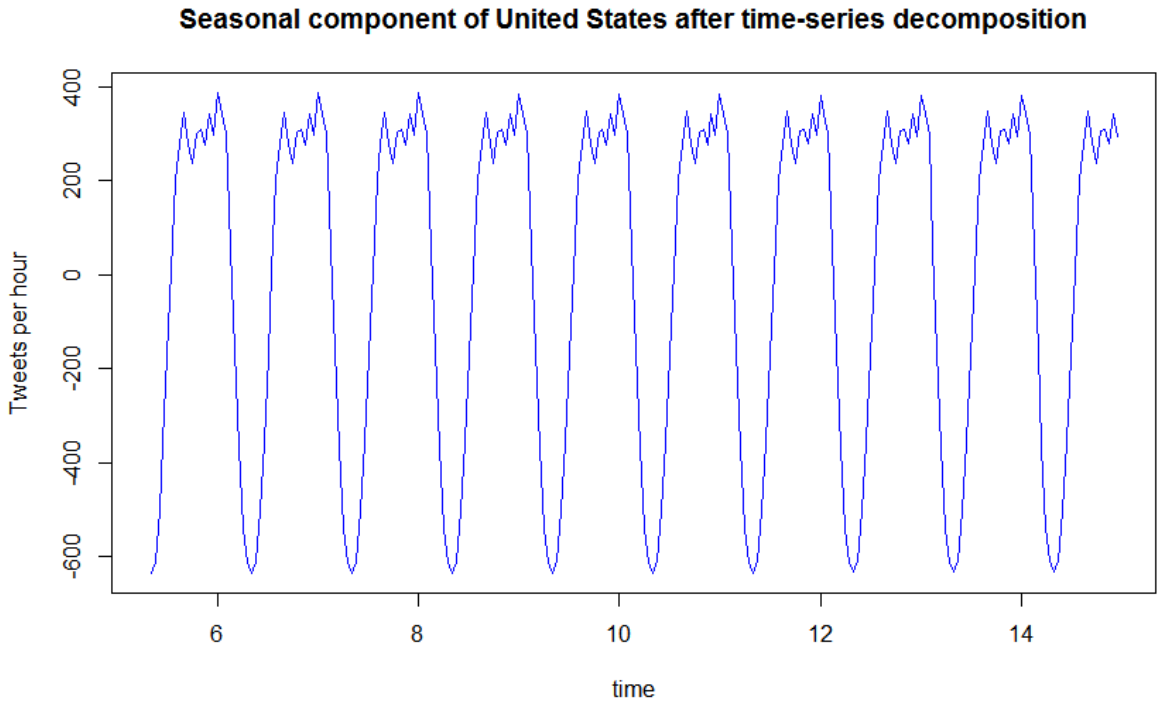


Figure 8 Seasonal component of U.S. after time-series decomposition

The range of y-axis of the seasonal component, which ranged to negative value, needs to be explained?: The seasonal component describes the seasonal impacts on the observation data, so that these impacts could be negative. For instance, the value seasonal component is around -600 tweets per hour at the late-night of U.S. local time, this does not mean that people post about -600 tweets at this hour daily (which is irrational), but means at this hour, the seasonal component impacts -600 tweets on the original data.

The contiguous U.S. is using four time zones ranged from UTC -5 to UTC -8. Here we defined four parts of a day at the UTC -7, which zone is the average of all time zones U.S.: morning (6:00 – 12:00), afternoon (12:00 – 18:00), night (18:00 – 0:00) and late-night (0:00 – 6:00). Then one cycle (24hour) of seasonal component could be shown in Figure 9,

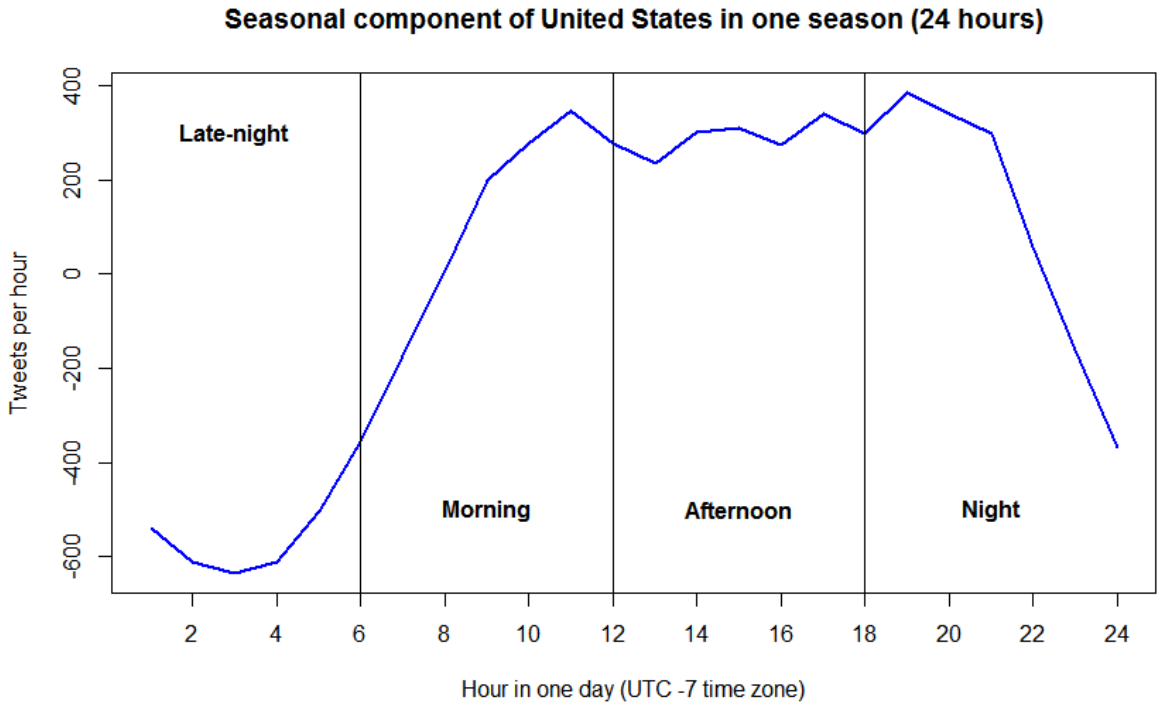


Figure 9 Seasonal component of U.S. in one day

We found that the seasonal component were highly related to the alternating of day and night. The seasonal component would be removed from the original data during the seasonal adjustment part, more discussion about further studies on seasonal component itself will be available in the discussion chapter.

4.4.2 Seasonal Adjustment

After decomposition of time-series, the seasonal adjustment was applied to remove the seasonal influence on the tweets data, the result was shown in the Figure 10,

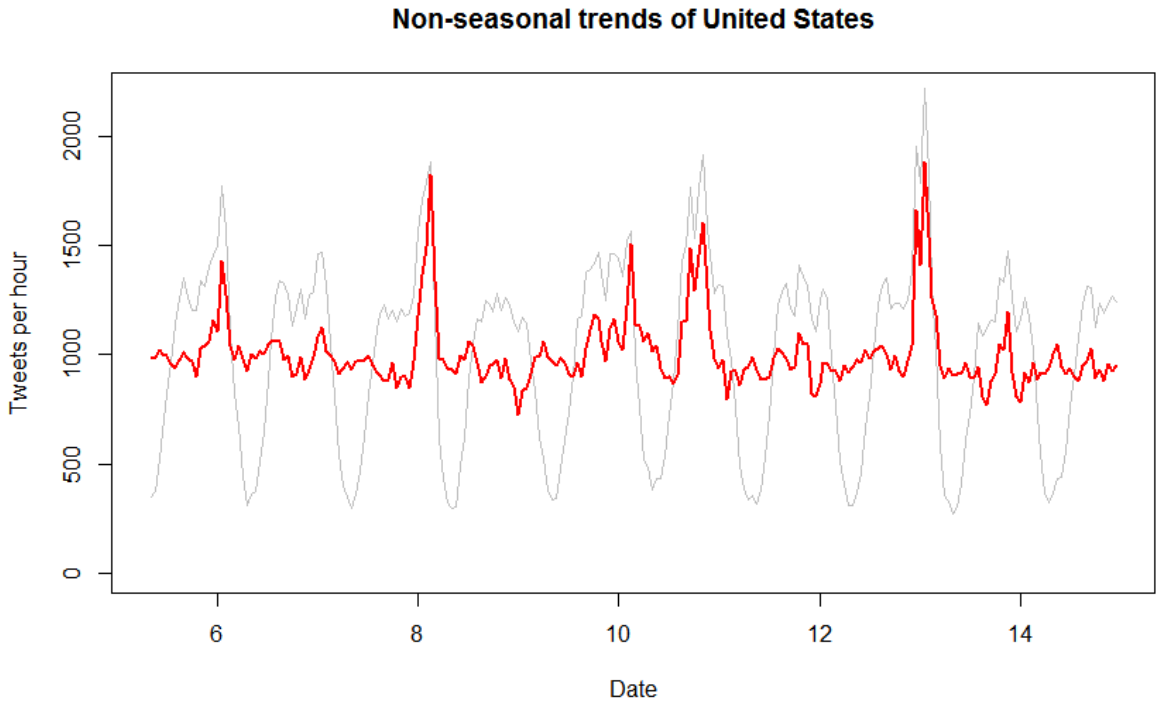


Figure 10 Non-seasonal trends of U.S. contrasting to the original data

The red line in Figure 10 is the non-seasonal trends of U.S. after seasonal adjustment, including the trending component and remainder component of the time-series decomposition. The grey line as background was the original data of U.S. before time-series decomposition and seasonal adjustment. We could read from this contrast that after seasonal adjustment, the extreme value of tweets were more apparent after removing the impacts from seasonal component and the seasonal minimal values were removed from the original data.

4.4.3 Recognise Pattern of Interests

After seasonal adjustment, we recognised the pattern of interests from the non-seasonal trends for Spatial-Temporal Events decomposition. The lagged differences were calculated on the non-seasonal trends with lag = 1. We manually set a threshold value 300: The lagged difference higher than 300 were treated as patterns of interests. The result was shown in the following Figure 11,

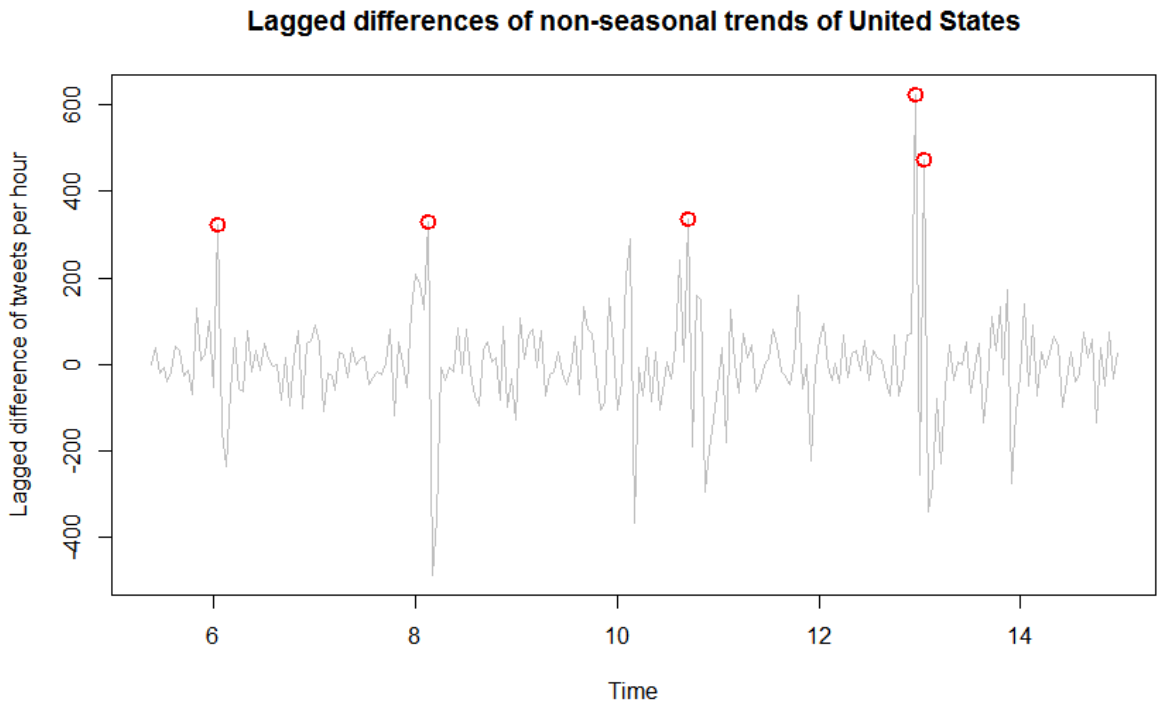


Figure 11 Lagged differences of non-seasonal trends of U.S.

Where red points indicated the pattern of interests detected from the lagged difference, retrieving these five points to the non-seasonal trends, we got Figure 12 as following,

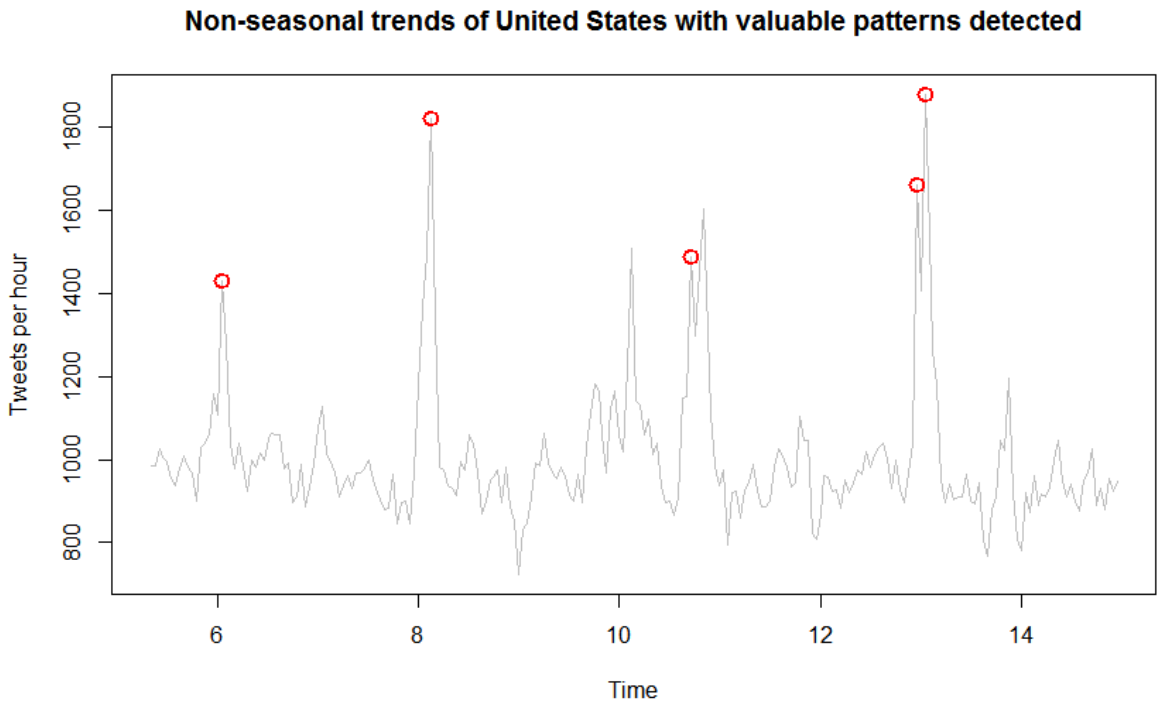


Figure 12 Pattern of interests in Non-seasonal trends of U.S.

These five pattern of interests were named pattern I to pattern V by the time sequence, the happening time of them were shown in the Table 9,

Pattern name	Time (approximate, in UTC +0)
Pattern I	2:00 of 6th Nov 2014 to 3:00 of 6th Nov 2014
Pattern II	4:00 of 8th Nov 2014 to 5:00 of 8th Nov 2014
Pattern III	18:00 of 10th Nov 2014 to 19:00 of 10th Nov 2014
Pattern IV	0:00 of 13th Nov 2014 to 1:00 of 6th Nov 2014
Pattern V	2:00 of 13th Nov 2014 to 3:00 of 13th Nov 2014

Table 9 Pattern of interests detected of U.S.

4.4.4 Detecting Spatial-Temporal Events

Manually scanning on tweets was applied on these patterns of interests: first we scanned all the tweets on that period with a location named in U.S. to find out a possible topic(s) that could cause the extreme value of the amount of related tweets. Here you will see the step by step analysis of Pattern I, and the brief result of Pattern II to V.

Pattern I

There are 1770 tweets related to U.S. posted at the time period of pattern I. Although we estimated the seasonal impact on the tweets data by time-series decomposition, we did not know which tweets belongs to the seasonal component. So here we use the tweets in location names detected dataset contains both seasonal component and non-seasonal trends.

All the location names mentioned in this period were calculate a ratio of total amount (1770 in this case). The top three result was shown in Table 10.

Location names	Ratio of total amount of tweets
Virginia (also VA, Richmond and Virginia Beach)	11.30%
U.S. (also USA, United States, US and America)	8.42%
New York (also New York City and NY)	8.36%

Table 10 Top three ratio of location names in Pattern I

There may be several Spatial-Temporal Events hidden in this pattern, but in this case we focused on the most interesting location name: Virginia, which even ranked higher than New York and United States that challenged our imagination. The composing of the topics of these tweets related to Virginia was manually scanned and shown in Table 11,

Topic	Related Tweets (Ratio of all tweets related to Virginia)
VA governor race	197 (98.5 %)
Other topics	3 (1.5 %)

Table 11 Ratio of topics of Tweets related to Virginia

Time-series decomposition and seasonal adjustment then was applied on all the tweets related to Virginia, the result was in the Figure 13,

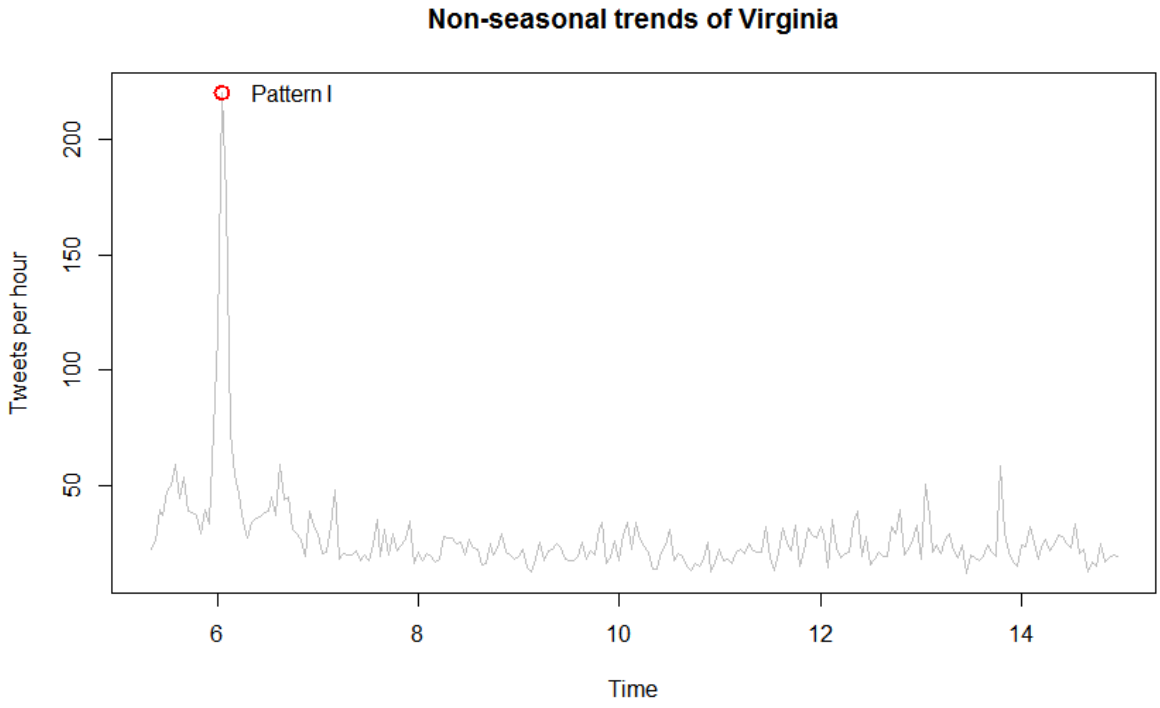


Figure 13 Non-seasonal trends of Virginia

A Spatial-Temporal Events was detected till now, with a temporal factor threshold 2:00 of 6th Nov 2014 to 3:00 of 6th Nov 2014, spatial factor threshold: Virginal Region which covered all location names of/in Virginal State and an event identifier: VA governor race. 197 tweets, as Spatial-temporal event occurrences, were existed in these thresholds.

The validation was applied on the Wikipedia and news of American websites: There was a Virginal Governor Race happened exactly on 5th of Nov 2013 (America local time), which is the same day as the Spatial-Temporal Events we detected from Pattern I (our temporal threshold was in UTC +0 Time zone). Breaking news of the result of this race came at 5th of Nov 2013 (America local time) and declarative news came since one days after 5th. This Spatial-Temporal Event we detected was existed and correct on time and position.

Using similar method, we manually scanned all five patterns, result was shown in following in brief.

Pattern II

Location names	Ratio of total amount of tweets
Oregon (also Portland)	36.49%
U.S. (also US, USA and America)	7.39%

Table 12 Top two ratio of location names in Pattern II

Then we select Oregon State to focus on,

Topic	Related Tweets (Ratio of all tweets related to Virginia)
Football game of Oregon	665 (96.94 %)
Other topics	21 (3.06 %)

Table 13 Ratio of topics of Tweets related to Oregon

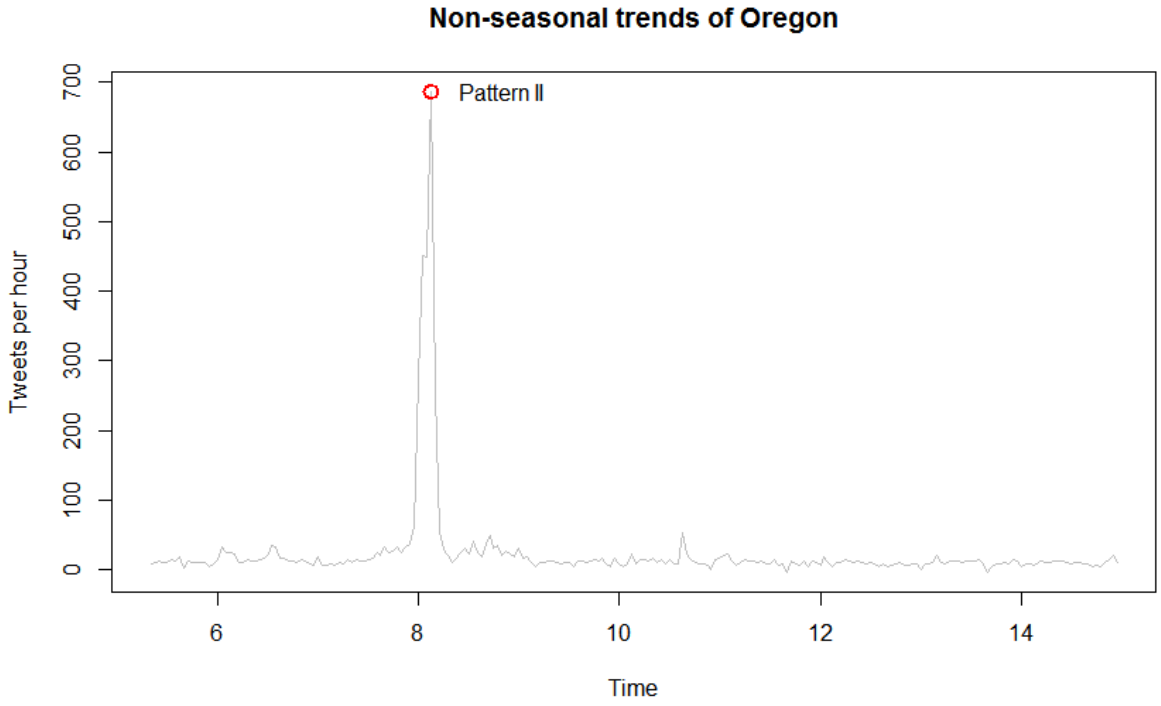


Figure 14 Non-seasonal trends of Oregon

The Spatial-Temporal Event was detected: temporal factor threshold 4:00 of 8th Nov 2014 to 5:00 of 8th Nov 2014, spatial factor threshold all location names in/of Oregon state, event identifier American football game Oregon Ducks vs. Stanford Cardinal. 665 Spatial-Temporal Event occurrences were in these thresholds.

Pattern III

Location names	Ratio of total amount of tweets
U.S. (also US, USA and America)	20.89%
Austin	16.70%

Table 14 Top two ratio of location names in Pattern III

69.65% of the tweets related to U.S. and almost 100% of the tweets related Austin were wrong results of location names detecting (Stanford NER model). There was an online voting happened at that period and one side of this voting was a singer named Austin. So this pattern was proved a failing detecting.

Pattern IV

Location names	Ratio of total amount of tweets
Kentucky	22.14%

Table 15 Top two ratio of location names in Pattern IV

Then we select Kentucky State to focus on,

Topic	Related Tweets (Ratio of all tweets related to Virginia)

NCAA[1] Basketball game of Kentucky State team	432 (99.77 %)
Other topics	1 (0.23 %)

Table 16 Ratio of topics of Tweets related to Kentucky

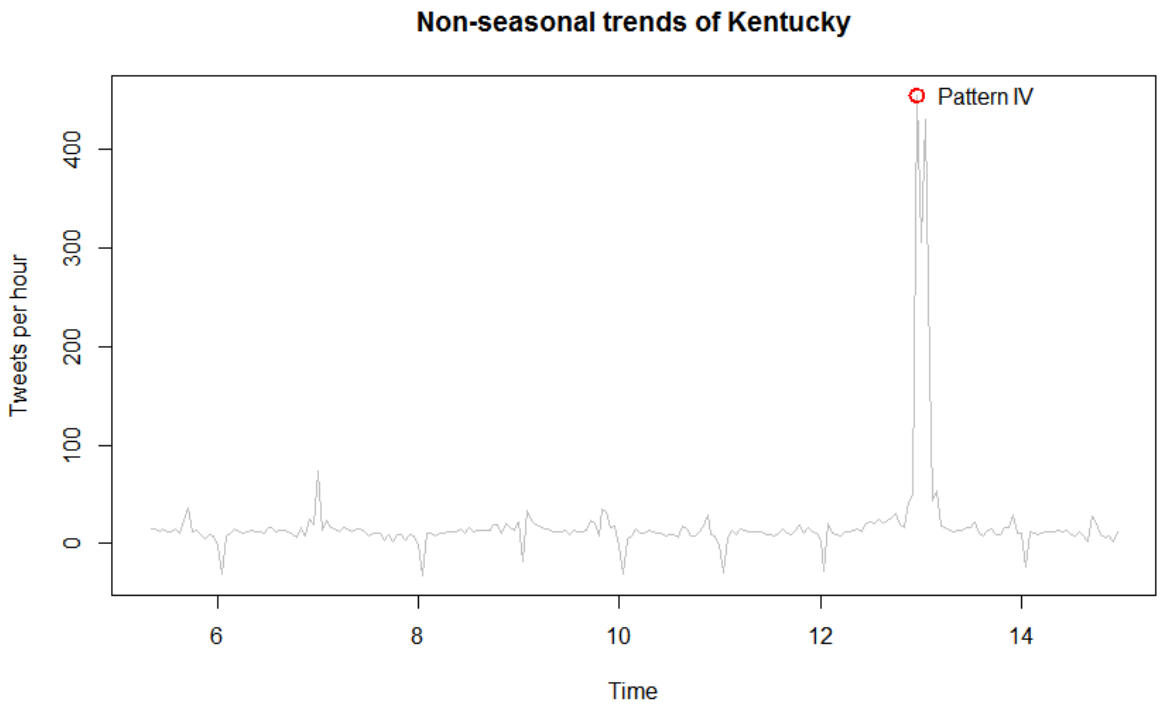


Figure 15 Non-seasonal trends of Kentucky (Pattern IV)

The Spatial-Temporal Event detected was: temporal factor threshold 0:00 of 13th Nov 2014 to 1:00 of 6th Nov 2014, spatial factor threshold Kentucky State, and event identifier NCAA basketball game Kentucky State vs. Michigan. 432 Spatial-Temporal Event occurrences (tweets) were in these thresholds.

Pattern V

Location names	Ratio of total amount of tweets
Kentucky	19.68%
U.S. (also US, USA and America)	15.50%

Table 17 Top two ratio of location names in Pattern V

Then we select Kentucky State to focus on,

Topic	Related Tweets (Ratio of all tweets related to Virginia)
NCAA[2] Basketball game of Kentucky State team	436 (99.77 %)

[1] NCAA: National Collegiate Athletic Association
[2] NCAA: National Collegiate Athletic Association

Other topics	1 (0.23 %)

Table 18 Ratio of topics of Tweets related to Kentucky

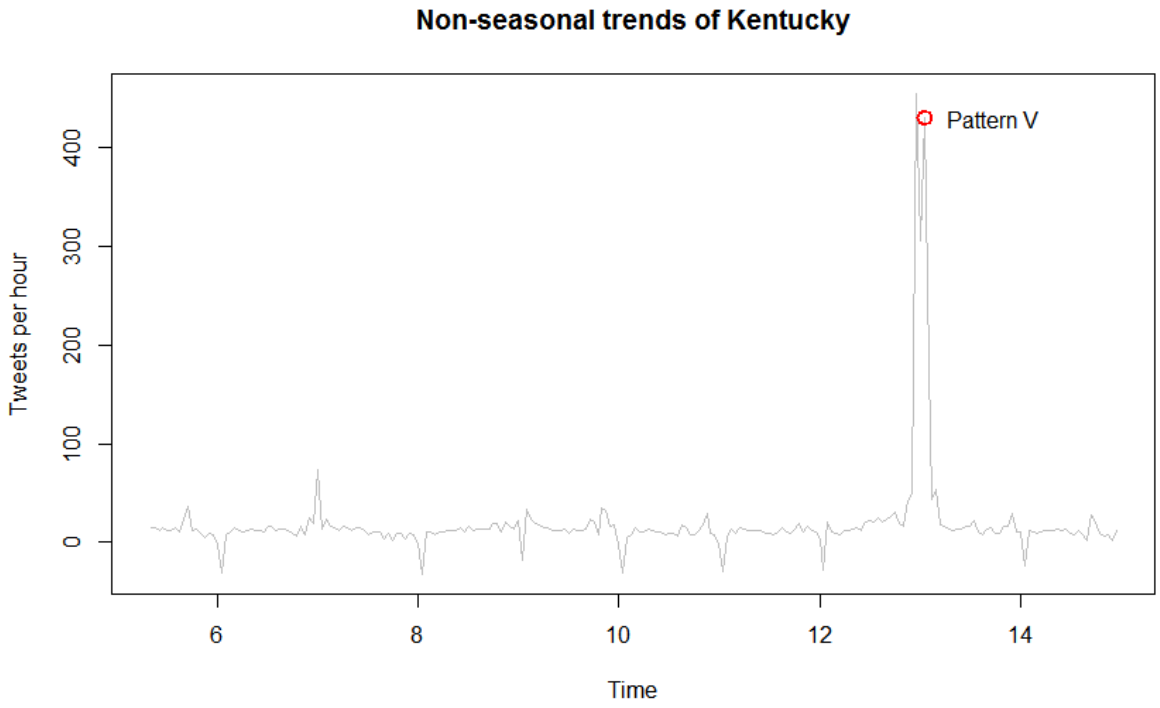


Figure 16 Non-seasonal trends of Kentucky (Pattern V)

The Spatial-Temporal Event detected was: temporal factor threshold 2:00 of 13th Nov 2014 to 3:00 of 6th Nov 2014, spatial factor threshold Kentucky State, and event identifier NCAA basketball game Kentucky State vs. Michigan. 436 Spatial-Temporal Event occurrences (tweets) were in these thresholds.

In summary, from these five pattern of interests, we detected Spatial-Temporal Events as Table 19,

Pattern	**Temporal factor threshold**	**Spatial factor threshold**	**Event identifier**	**Amount of event occurrences (Tweets)**
I	2:00 of 6th Nov 2014 to 3:00 of 6th Nov 2014	Virginia State	VA governor race	197
II	4:00 of 8th Nov 2014 to 5:00 of 8th Nov 2014	Oregon State	Football game of Oregon	665
III	failed	failed	failed	failed
IV	0:00 of 13th Nov 2014 to 1:00 of 6th Nov 2014	Kentucky State	NCAA Basketball game of Kentucky State team	432

V	2:00 of 13th Nov 2014 to 3:00 of 13th Nov 2014	Kentucky State	NCAA Basketball game of Kentucky State team	436

Table 19 Spatial-Temporal Events detected from patterns of interests

The Spatial-Temporal Event detected from Pattern IV and V, are two Spatial-Temporal Event shared same spatial factor and event identifier but different temporal factor, if we extend the temporal to three hours, then they could be combined into one Spatial-Temporal Event, the final result after combination was shown in the Table 20, and all of them were validated based on the news and match records.

ID	Temporal factor threshold	Spatial factor threshold	Event identifier	Amount of event occurrences (Tweets)	Detected from
1	2:00 of 6th Nov 2014 to 3:00 of 6th Nov 2014	Virginia State	VA governor race	197	Pattern I
2	4:00 of 8th Nov 2014 to 5:00 of 8th Nov 2014	Oregon State	Football game of Oregon	665	Pattern II
3	0:00 of 13th Nov 2014 to 3:00 of 13th Nov 2014	Kentucky State	NCAA Basketball game of Kentucky State team	1167	Pattern IV, V, and the part between them

Table 20 Spatial-Temporal Events detected after combination

CHAPTER 5: DISCUSSION

This thesis developed a methodology for detecting Spatial-Temporal Events from tweets data. Based on the definition of Spatial-Temporal Events, the spatial factor, temporal factor and event identifier were extracted by integrating various methods:

- The temporal factor was originally gained during the data collection. The spatial factor was extracted by the Stanford NER model, which was different from the traditional method: Geo-targeting. This Stanford NER model proved to be able to analyse tweets data and extract location names in tweets for Geo-information research. One important advantage of this model contrasting to geo-targeting is that the location names were directly extracted from texts. For the new era of big data, the class of NER models provide a possibility to collect location information beyond the supporting from hardware equipment such as GPS or IP address. Especially for the information on social network service, the Geo-targeting needs user's equipment supporting, the IP address cannot be widely available since it potentially violates the user's privacy, only the texts posted on social network are widely opened to public and researchers. Applying the Stanford NER model on these texts to extract spatial factor prospectively allow the data enter the model and avoid the shortcomings of GPS or IP address.
- Comparing to traditional keyword-driven research about events and twitter, this thesis tried to go beyond the use of simple keywords. Although the keywords were still used at the extracting of the topic of Spatial-Temporal Events (event identifier), this research filtered and grouped both the spatial factor and temporal factor before identifying the topic/keywords of Spatial-Temporal Events.

  This means that during the processing on spatial factor and temporal factor, the topic of the Spatial-Temporal Event caused the changes on spatial and temporal factor can be unknown. First, we were aware of the noticeable changes on spatial and temporal factor (so called patterns of interests), then the topic/keywords were extracted. This makes the detecting of Spatial-Temporal Events could fit the keywords-unknown monitoring on tweets. In some possible situations, at data level, this method can drive the researcher to monitor the likelihood of Spatial-Temporal Events without knowing which event it is.
- To be aware of the relations among spatial factor, this research used a hierarchy of location names instead of coordinates. In theory, the coordinates is able to solve all the problems of the relations among different location names. However, due to the size and structure of data, the hierarchized location names by administrative levels become an alternative method for the use of coordinates. This hierarchy simplified the understanding of these relations in this research, for the analysis on location names, it can be a good method for clearing data before the next steps.

The main conditions which limit this thesis is the stress on human resource and time. If larger datasets and entire analysing steps are allowed on this research, it can be improved on both theoretical and practical aspects in following ways:

- Automatize the manual works. The lack of statistical features of different location names lead this research to rely on manual works, to achieve these features, more analysis could be applied on tweets data. For instance, apply a further analysis on seasonal component. Even though we removed the seasonal component during the seasonal adjustment of the time-series to reduce the impact of regular component of tweets, the seasonal component may tell the statistical features of a given location name.

- Improve the Stanford NER model and its validation. In this thesis, we used the Stanford NER trained by pre-defined training set, yet if the time and human resource allowed, a way to provide more accurate NER model is to train the model based on the tweets data. The pre-defined training sets only focused on the general rules of natural language, the feature of language using in tweets, for example the abbreviations and spelling habits, could be studied by training NER model with tweets dataset. A good example is a typical error we mentioned in the last chapter: “US” was wrongly recognised as “United States”, indeed it was only the upper case of English word “us”. This is because in the normal sentences on article, newspapers and books, people hardly spell “us” in upper case, but in tweets, it may be a custom that people have a tendency to spell some words, even whole sentence in upper case to describe their intense attitude.

  The validation of Stanford NER model, which was pointed out in chapter 4 that could be done by hypothesis testing, would be manually executed if the time and human resource allowed.

- A flexible temporal factor threshold. In this research, the data was collected in the time scale “second” then converted to the time scale “hour”. The temporal factor threshold of Spatial Temporal Events was set to one hour based on this conversion. Further analysis of temporal factor threshold could be applied on the change of scale and the beginning time of this conversion.

  The scale of temporal factor has not to be one hour, it can range from minute to day. And the beginning time of the conversion from second to minute/hour/day is not necessary to be fixed on the first second. For example, for the scale of hour, the range of one hour could be 00:00:01 to 01:00:00, but it could also be 00:00:02 to 01:00:01 and so on. The changing of beginning second can be described as a moving time window with a fixed width such as minute/hour/day. Extending on different size and beginning time of temporal factor threshold offers entire analysis on temporal factor so that more patterns of interests may be extracted than the current one.

CHAPTER 6: CONCLUSION

The thesis research developed a semi-automatic model to detect Spatial-Temporal Events from tweets data. Three Spatial-Temporal Events in U.S. were detected from the case study and these events were validated based on the internet news. The validation based on internet news proved that the series of approaches used in this thesis successfully detected Spatial-Temporal Events from Twitter data. The Twitter data could be used for detecting Spatial-Temporal Events, even though some steps had to be done manually, the possibility of detecting Spatial-Temporal Events automatically would be expectable after further studies.

The research questions presented at the beginning of this report now can be answered as following points:

- The Spatial-Temporal Events were defined as a set of Spatial-Temporal Event occurrences with spatial factor, temporal factor and event identifier. In mathematics, it was defined as a tuple of three elements. Based on three assumptions, tweets successfully presented Spatial-Temporal Event occurrences, which built the fundamental of detecting Spatial-Temporal Event from Twitter data.
- The spatial factor is the latent location names in tweets. They were extracted by Stanford NER model. This was thought a novel method to extract spatial factor from tweets without geo-targeting.
- The location names extracted by Stanford NER were converted to coordinates by geo-coding. The Geo-coding also hierarchized these location names based on administrative levels to be aware of the relations among them. This hierarchy of locations worked well and took the place of coordinates in this thesis, because it avoided the complication may occurred by coordinates.
- The amount of located tweets in time sequence was selected as a characteristic of tweeting behaviour for detecting Spatial-Temporal Events. This characteristic was made more notable after time-series decomposition and seasonal adjustment. The extreme changes of this characteristic were considered as patterns of interests, which were found manually with the help of lagged difference.
- The Spatial-Temporal Events were detected by manually checking on the tweets in these patterns of interests. In the five patterns of interests detected, four of them were confirmed containing a Spatial-Temporal Event inside.
- The detected Spatial-Temporal Events were validated from related news at the detected time and location, yet the validation of Stanford NER model was not fully made since the limit on time and human resource.

REFERENCES

Abel, F., Hauff, C., Houben, G.-J., Stronkman, R., & Tao, K. (2012). Twitcident: fighting fire with information from social web streams. In *Proceedings of the 21st international conference companion on World Wide Web* (pp. 305–308).

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python*. O'Reilly Media, Inc.

Cataldi, M., Caro, L. Di, & Schifanella, C. (2010). Emerging topic detection on Twitter based on temporal and social terms evaluation. *Proceedings of the Tenth …*.

Cha, M., Haddadi, H., Benevenuto, F., & Gummadi, P. (2010). Measuring User Influence in Twitter: The Million Follower Fallacy. *ICWSM*, 10–17.

Chew, C., & Eysenbach, G. (2010). Pandemics in the age of Twitter: content analysis of Tweets during the 2009 H1N1 outbreak. *PloS One*, *5*(11), e14118. doi:10.1371/journal.pone.0014118

Cleveland, R. B., Cleveland, W. S., McRae, J. E., & Terpenning, I. (1990). STL: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics*, *6*(1), 3–73.

Dollár, P., Rabaud, V., Cottrell, G., & Belongie, S. (2005). Behavior recognition via sparse spatio-temporal features. In *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on* (pp. 65–72).

Finin, T., Murnane, W., & Karandikar, A. (2010). Annotating named entities in Twitter data with crowdsourcing. *… and Language Data …*, *2010*(June), 80–88.

Finkel, J. R., Grenager, T., & Manning, C. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics* (pp. 363–370).

Fleischman, M. (2001). Automated subcategorization of named entities. *ACL (Companion Volume)*.

Guralnik, V., & Srivastava, J. (1999). Event detection from time series data. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 33–42).

Kisilevich, S., Krstajic, M., Keim, D., Andrienko, N., & Andrienko, G. (2010). Event-based analysis of people's activities and behavior using Flickr and Panoramio geotagged photo collections. In *Information Visualisation (IV), 2010 14th International Conference* (pp. 289–296).

Lafferty, J., McCallum, A., & Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Lauw, H. W., Lim, E.-P., Pang, H., & Tan, T.-T. (2005). Social Network Discovery by Mining Spatio-Temporal Events. *Computational and Mathematical Organization Theory*, *11*(2), 97–118. doi:10.1007/s10588-005-3939-9

Lee, C.-H. (2012). Mining spatio-temporal information on microblogging streams using a density-based online clustering method. *Expert Systems with Applications*, *39*(10), 9623–9641.

Liu, X., Zhang, S., Wei, F., & Zhou, M. (2011). Recognizing Named Entities in Tweets. *ACL*, (2008), 359–367.

Locke, B., Martin, J., & Ph, D. (2009). Named Entity Recognition : Adapting to Microblogging Named Entity Recognition , Adapting to Microblogging. *University of Colorado*.

Lunden, I. (2012). Analyst: Twitter Passed 500M Users In June 2012, 140M of Them in US; Jakarta “Biggest Tweeting”City. Thesis. AOL Tech, 2012. Tech Crunch.

Mathioudakis, M., & Koudas, N. (2010). Twittermonitor: trend detection over the twitter stream. *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, 1155–1157.

Nadeau, D., & Sekine, S. (2007). A survey of named entity recognition and classification. *Lingvisticae Investigationes*, (1991), 1–20.

Niebles, J. C., Wang, H., & Fei-Fei, L. (2008). Unsupervised learning of human action categories using spatial-temporal words. *International Journal of Computer Vision*, *79*(3), 299–318.

Okazaki, M., & Matsuo, Y. (2011). Semantic Twitter : Analyzing Tweets for Real-Time Event Notification. In *Recent Trends and Developments in Social Software* (pp. 63–74). Springer.

Ratinov, L., & Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning* (pp. 147–155).

Rau, L. F. (1991). Extracting company names from text. In *Artificial Intelligence Applications, 1991. Proceedings., Seventh IEEE Conference on* (Vol. i, pp. 29–32). doi:10.1109/CAIA.1991.120841

Ritter, A., Clark, S., & Etzioni, O. (2011). Named entity recognition in tweets: an experimental study. *… of the Conference on Empirical Methods …*, 1524–1534.

Sakaki, T., Okazaki, M., & 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).

Sundheim, B., Road, G., Diego, S., Grishman, R., & York, N. (n.d.). Message U n d e r s t a n d i n g C o n f e r e n c e - 6 : A Brief History Ocean Surveillance Center Evaluation Division (NRaD) Short-term subtasks Portability.

Tr-, T. R., Labs, M., Zhao, S., & Zhong, L. (2011). Human as Real-Time Sensors of Social and Physical Events : A Case Study of Twitter and Sports Games, (June), 1–9.

Twitter, O. B. (2011). 200 million tweets per day. *Retrived at August 17th*.

Vieweg, S., Hughes, A. L., Starbird, K., & Palen, L. (2010). Microblogging during two natural hazards events: what twitter may contribute to situational awareness. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 1079–1088).

White, J., & Roth, R. (2010). TwitterHitter: Geovisual analytics for harvesting insight from volunteered geographic information. *Proceedings of GIScience*.

Yin, J., Hu, D. H., & Yang, Q. (2009). Spatio-Temporal Event Detection Using Dynamic Conditional Random Fields. In *IJCAI* (Vol. 9, pp. 1321–1327).

Zhang, R. J., & Unger, E. A. (1996). Event Specification and Detection.

APPENDIX

APPENDIX A: PYTHON CODE FOR CATCHING TWEETS FROM TWITTER STREAMING

Python version: v2.7.5 ab05e7dd2788

Operating system version: OS X Mavericks 10.9.2 (These codes are also suitable for Windows users)

```
#Receiving Tweets from Twitter streaming
#Thesis work by LIU Yue at Wageningen University as a MGI student.
#version 1.0 is made to get the real-time streaming data of twitter and
store them into xls file format

__author__ = 'yue.liu@wur.nl'
__version__ = '1.0'

from tweepy.streaming import StreamListener
from tweepy import OAuthHandler
from tweepy import Stream
#tweepy is an open-source plugin package for processing Tweets on Python
from xlwt import *
import time

consumer_key="-"
consumer_secret="-"

access_token="-"
access_token_secret="-"
#Those are private keys, which can be created by registered user on
#http:\\dev.twitter.com
#They were removed while be displayed on thesis report since the security
#reason

class StdOutListener(StreamListener):

    #Here we created a listener that just response received tweets then
store them.
    def __init__(self):
        self.rowcount = 0
        self.filenumber = 0
        self.wb = Workbook()
        self.ws = self.wb.add_sheet('0')
    def on_data(self, data):
        pos = data.rfind("\"lang\"") + 8
        #data begin with {""created_at""
        #                01234567890123
        #for some deleted tweet, it will not begin with "created_at", which
#will be filterd

        if data[pos:pos+2] == "en":   #Only keep English Tweets
            if(self.rowcount < 20000):
                #data format: "created_at":"Mon Oct......","id"
                #             012345678901234567
                pos1 = data.find("\"created_at\"")+14
                pos2 = data.find("\"id\"") - 2
                creat_time = data[pos1:pos2]
                self.ws.write(self.rowcount,0,creat_time)
                #data format ..."text":"Hello world!","source""
                #               01234567890
                pos1 = data.find("\"text\"") + 8
```

```
                pos2 = data.find("\"source\"") - 2
                creat_text = data[pos1:pos2]
                self.ws.write(self.rowcount,1,creat_text)
                self.rowcount += 1
            else:
                filetime = time.strftime("%m_%d_%H_%M_%S")
                filename = str(self.filenumber) +
"tweetdata"+filetime+".xls"
                self.wb.save(filename)
                self.filenumber += 1
                self.wb = Workbook()
                self.ws = self.wb.add_sheet('0')
                self.rowcount = 0
                print str(self.filenumber) + "finish"

        return True

    def on_error(self, status):
        print status

if __name__ == '__main__':

    test_listener = StdOutListener()
    auth = OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)
    stream = Stream(auth, test_listener)
    stream.sample()
    #stream.sample() provides a random sample from the current tweets
#streaming
    #This random sample will enter the listener then be filtered and
#stored
```

APPENDIX B: JAVA CODE OF INPUTTING TWEETS DATA INTO STANFORD NER MODEL

Java environment: Eclipse Standard/SDK Kepler Release 20130614-0229.

Operating system: OS X Mavericks 10.9.2

```
//Stanford NER
//Amended by yue.liu@wur.nl to fit his dataset for thesis in Wageningen UR

//import class from Stanford NER
import edu.stanford.nlp.ie.AbstractSequenceClassifier;
import edu.stanford.nlp.ie.crf.*;
import edu.stanford.nlp.io.IOUtils;
import edu.stanford.nlp.ling.CoreLabel;
import edu.stanford.nlp.ling.CoreAnnotations;
//jxl is used for access xls file
import jxl.*;
import jxl.read.biff.BiffException;
import jxl.write.Label;
import jxl.write.WritableSheet;
import jxl.write.WritableWorkbook;
import jxl.write.WriteException;
import jxl.write.biff.RowsExceededException;

import java.util.List;
import java.io.File;
import java.io.IOException;

public class NER_test {

    @SuppressWarnings("null")
    public static void main(String[] args) throws IOException {

      String serializedClassifier =
"classifiers/english.all.3class.distsim.crf.ser.gz";
      //This is the classifier we used in this thesis

      if (args.length > 0) {
        serializedClassifier = args[0];
      }

      AbstractSequenceClassifier<CoreLabel> classifier =
CRFClassifier.getClassifierNoExceptions(serializedClassifier);

      if (args.length > 1) {
        String fileContents = IOUtils.slurpFile(args[1]);
        List<List<CoreLabel>> out = classifier.classify(fileContents);
        for (List<CoreLabel> sentence : out) {
          for (CoreLabel word : sentence) {
            System.out.print(word.word() + '/' +
word.get(CoreAnnotations.AnswerAnnotation.class) + ' ');
          }
          System.out.println();
        }
        out = classifier.classifyFile(args[1]);
        for (List<CoreLabel> sentence : out) {
          for (CoreLabel word : sentence) {
            System.out.print(word.word() + '/' +
word.get(CoreAnnotations.AnswerAnnotation.class) + ' ');
          }
          System.out.println();
        }
```

```
      } else {
        int[] my_array = new
int[]{0,27,77,126,174,224,285,336,386,435,483,516,562,613,664,691};
        //This set of numbers is pre-defined for produce correct file
number: ,fn

        for(int d = 1; d<my_array.length;d++){


        for(int fn = my_array[d-1]; fn< my_array[d]; fn++){
            int tt = 4+d;
            String st = new String();
            if (tt< 10) st = "0" + Integer.toString(tt);
            else st = Integer.toString(tt);
            String xlsname = "raw_tweetdata13_11_" + st +"_fn"
+Integer.toString(fn) +".xls";

        try {
            //create a new worksheet to get data from previous
dataset(wrk1.sheet1) and save data in new file(wkr2.sheet2)
            Workbook wrk1 =  Workbook.getWorkbook(new
File("/Users/liuyue/Documents/workspace/data/"+xlsname));
            Sheet sheet1 = wrk1.getSheet(0);
            WritableWorkbook wrk2 = Workbook.createWorkbook(new
File("/Users/liuyue/Documents/workspace/data/NER_RESULT/" +
"NER_"+xlsname));
            WritableSheet sheet2 = wrk2.createSheet("Sheet1", 0);
            //Obtain reference to <span id="IL_AD10" class="IL_AD">the
Cell</span> using getCell(int col, int row) <span id="IL_AD9"
class="IL_AD">method</span> of sheet
            int i = 0;
            for(i= 0;i<30000;i++){
            Cell datetime = sheet1.getCell(0,i);
            String tweettime = datetime.getContents();
            Cell colrow = sheet1.getCell(1, i);
            String tweet = colrow.getContents();
            String sentence = new String();
            sentence = classifier.classifyWithInlineXML(tweet); // The text
will be classified to inline XML structure
            String[] location_name = new String[30];

            int num_location = 0; //how many location names that this tweet
has.
            //extract location from named entity
            while (sentence.indexOf("<LOCATION>") > -1){
                int mm = sentence.indexOf("<LOCATION>");
                mm = mm + 10;
                int nn = sentence.indexOf("</LOCATION>");
                String t_location = sentence.substring(mm,nn);
                location_name[num_location] = t_location;

                sentence = sentence.substring(nn+11);
                num_location++;
            }//end of while


            Label lb1 = new Label(0,i,tweettime);
            //System.out.println(lb1.getString());
            Label lb2 = new Label(1,i,tweet);
            Label[] lb = new Label[30];
```

```
            for(int lo = 0; lo < num_location; lo++){
                int num_col = 2 + lo;
                lb[lo] = new Label(num_col,i,location_name[lo]);
            }
            //Label lb3 = new Label(2,i,location_name);
            try {
                sheet2.addCell(lb1);
                sheet2.addCell(lb2);
                for(int sheet_i = 0; sheet_i < num_location;sheet_i++){
                    sheet2.addCell(lb[sheet_i]);
                }

            } catch (RowsExceededException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            } catch (WriteException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }


            }
            try {
                wrk2.write();
                wrk2.close();
            } catch (WriteException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
            wrk1.close();
            //Read the contents of the Cell using getContents() method,
which will return
            } catch (BiffException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        //System.out.println(classifier.classifyWithInlineXML(s2));
        //System.out.println(classifier.classifyToString(s2, "xml", true));

      }
      }
      }
    }
}
```

APPENDIX C: EXAMPLE OF DATASET AFTER STANFORD NER PROCESSING

Time	Location	Tweet
Thu Nov 07 17:48:58 +0000 2013	mindanao	please be safe guys. pray lng. esp. yung nasa visayas at mindanao area. Stay dry. #PrayForThePhilippines #YolandaPH
Thu Nov 07 17:49:01 +0000 2013	US	Breaking News: US stocks: Wall St lower, Twitter sparkles in debut http:\/\/t.co\/rlsPr2aN25
Thu Nov 07 17:49:01 +0000 2013	Baltimore	@cwgabriel just move PAX East to Baltimore.
Thu Nov 07 17:49:01 +0000 2013	Delhi	RT @alok_bhatt: Dear @rajuparulekar & @sureshpathare, onus lies on both of u to save Delhi voters from wasting their votes on a fraud like \u2026
Thu Nov 07 17:49:03 +0000 2013	Cleveland	I'm so old, I remember when WMMR was mad at Hall and Oates for supporting Cleveland over Philadelphia for Rock 'n' Roll Hall of Fame.
Thu Nov 07 17:49:03 +0000 2013	Philadelphia	I'm so old, I remember when WMMR was mad at Hall and Oates for supporting Cleveland over Philadelphia for Rock 'n' Roll Hall of Fame.
Thu Nov 07 17:49:04 +0000 2013	Austin	\u201c@so_URbwhite: @i_love_erica I need to eat with you !!!\u201d\u2022 I'm in Austin \ud83d\ude14
Thu Nov 07 17:49:05 +0000 2013	Argentina	@justinbieber Welcome To Argentina Bieber\u2665 We love you here :)
Thu Nov 07 17:49:08 +0000 2013	UK	RT @FilmOnLive: http:\/\/t.co\/FX4YtjuD47 is LIVE now in the UK on Sky Channel 292! Check us out!! #FilmOn #Entertainment #FilmOnTV http:\/\/t.\u2026
Thu Nov 07 17:49:13 +0000 2013	U.S.	The now-infamous Obamacare website cost U.S. taxpayers over $1 billion to build. #Obamacare
Thu Nov 07 17:49:13 +0000 2013	Dubai	Wanna go to Dubai
Thu Nov 07 17:49:17 +0000 2013	american	american horror story just gets better and better each week
Thu Nov 07 17:49:17 +0000 2013	Austin	RT @AboveMahomie: RT or Austin won't #BangaBanga you. #VoteAustinMahone
Thu Nov 07 17:49:22 +0000 2013	P.E	Football in P.E is just long balls and ridiculously optimistic shooting
Thu Nov 07 17:49:23 +0000 2013	California	RT @Brunocerous: Billionaire Closes Off Access to One of California's Best Public Beaches http:\/\/t.co\/OHJYjnYMv1 #ows
Thu Nov 07 17:49:23 +0000 2013	UK	RT @9ja_Ninja: Popular UK Queen Insult Nigerians (LOOK) http:\/\/t.co\/jhlcUQ5TJF #9jaNinjaDotCom
Thu Nov 07 17:49:25 +0000 2013	Italy	I did not go to the concert of May in Italy and I will not go even to that of June and July,u can at least follow me? thanks @onedirection 4
Thu Nov 07 17:49:31 +0000 2013	Nigeria	#ResilientAfrica @VenturesAfrica Nigeria Encourages Increased Female Participation In ICT http:\/\/t.co\/PL10yFmnan #USAID

Thu Nov 07 17:49:32 +0000 2013	US	Just commented on @thejournal_ie: Love\/Hate US remake confirmed, to be announced \u2018in weeks\u2019 - http:\/\/t.co\/nfizY98PoA
Thu Nov 07 17:49:32 +0000 2013	London	NEW HQ PIC: Harry out in London today (Nov 7) #3 http:\/\/t.co\/Us1662g5gl
Thu Nov 07 17:49:35 +0000 2013	England	How lampard is in the England squad and mark noble isn't is beyond me? I thought he's meant to be picking players in form lampard....
Thu Nov 07 17:49:36 +0000 2013	Lolll	I'm not trynna put pants on to go outside Lolll
Thu Nov 07 17:49:37 +0000 2013	Rome	It's funny cause I'm seeing Muse live in Rome but I'm NOT EVEN IN ROME
Thu Nov 07 17:49:37 +0000 2013	ROME	It's funny cause I'm seeing Muse live in Rome but I'm NOT EVEN IN ROME
Thu Nov 07 17:49:38 +0000 2013	Philip	Sean Og O hAilpin will be in Philip's Bookshop this Saturday at 3.30!! @corkgaa @seanogohailpin
Thu Nov 07 17:49:40 +0000 2013	Los Angeles	@Michael5SOS fly to Los Angeles NOW
Thu Nov 07 17:49:41 +0000 2013	Austin	@AustinMahone #voteaustinmahone #voteaustinmahone I'm so happy I can tweet this all day for Austin. Even though I'm home sick. \u2764\ufe0f
Thu Nov 07 17:49:41 +0000 2013	London	Nice to meet The Mad Hatter! Look for the new acquaintances in the streets of London! http:\/\/t.co\/vlxCU3GTUX #iPad #iPadGame...
Thu Nov 07 17:49:42 +0000 2013	Beverly Hills	I just listed: 'Beverly Hills Cop III: Original Motion Picture Soundtrack', for $0.20 via @amazon http:\/\/t.co\/BFOo6cQ1oE
Thu Nov 07 17:49:44 +0000 2013	London	I just spent \u00a3180 for a bed in London for one night next week #thisisnotok
Thu Nov 07 17:49:45 +0000 2013	Taeyang	RT @HapyVirus: Taeyang's rapping skill \u2665\u2665\u2665\u2665\u2665
Thu Nov 07 17:49:46 +0000 2013	Chile	Should of brung something too eat Chile
Thu Nov 07 17:49:46 +0000 2013	Oldham	@SamHeatley @BenTilTen transferred to Oldham uni, newcastle just wasn't for me
Thu Nov 07 17:49:47 +0000 2013	Daffa Room	Chill out ~ (with Muhamad at Daffa Room) \u2014 https:\/\/t.co\/pFAHRoQesa
Thu Nov 07 17:49:51 +0000 2013	Arizona	Al Qaeda 'Gleeful' Over Snowden Leaks, MI6 Head Says http:\/\/t.co\/fpgERbGcBL Riiiight. And I've got beach-front property in Arizona.
Thu Nov 07 17:49:53 +0000 2013	Durango Mountain	New images: Jacob and Janie\u2019s Durango Mountain Resort Wedding: \u00a0 Jacob and Janie\u2019s Durango Mountain Resort Wed... http:\/\/t.co\/IRF6WxZeXM
Thu Nov 07 17:49:53 +0000 2013	Durango Mountain	New images: Jacob and Janie\u2019s Durango Mountain Resort Wedding: \u00a0 Jacob and Janie\u2019s Durango Mountain Resort Wed... http:\/\/t.co\/IRF6WxZeXM

APPENDIX D: EXAMPLE OF GEO-CODED AND HIERARCHIZED LOCATION NAMES

Location Name	Latitude	Longitude	city	city_short	province	province_short	country	country_short
US	37.09024	-95.7129					United States	US
London	51.50852	-0.12549	London	London			United Kingdom	GB
Philippines	12.87972	121.774					Philippines	PH
UK	55.37805	-3.43597					United Kingdom	GB
Argentina	-38.4161	-63.6167					Argentina	AR
Austin	30.267115	-97.7431	Austin	Austin	Texas	TX	United States	US
America	37.09024	-95.7129					United States	US
New York	40.71435	-74.006	New York	New York	New York	NY	United States	US
Texas	31.9686	-99.9018			Texas	TX	United States	US
U.S.	37.09024	-95.7129					United States	US
India	20.59368	78.96288					India	IN
China	35.86166	104.1954					China	CN
Mexico	23.6345	-102.553					Mexico	MX
Australia	-25.2744	133.7751					Australia	AU
England	52.35552	-1.17432			England	England	United Kingdom	GB
Brazil	-14.235	-51.9253					Brazil	BR
Chicago	41.87811	-87.6298	Chicago	Chicago	Illinois	IL	United States	US
Canada	56.13037	-106.347					Canada	CA
Toronto	43.65323	-79.3832	Toronto	Toronto	Ontario	ON	Canada	CA
Sachin Tendulkar	19.05967	72.82332	Mumbai	Mumbai	Maharashtra	MH	India	IN
USA	37.09024	-95.7129					United States	US
Japan	36.20482	138.2529					Japan	JP
Florida	27.66483	-81.5158			Florida	FL	United States	US
Miami	25.78897	-80.2264	Miami	Miami	Florida	FL	United States	US
Oregon	43.80413	-120.554			Oregon	OR	United States	US
Chile	-35.6751	-71.543					Chile	CL
California	36.77826	-119.418			California	CA	United States	US
Europe	54.52596	15.25512						
Alabama	32.31823	-86.9023			Alabama	AL	United States	US
Paris	48.85661	2.352222	Paris	Paris	Île-de-France	IDF	France	FR
Michigan	44.31484	-85.6024			Michigan	MI	United States	US
Iran	32.42791	53.68805					Iran	IR

APPENDIX E: R CODE FOR TIME-SERIES DECOMPOSITION AND SEASONAL ADJUSTMENT

R version: 3.0.2 (2013-09-25)

Operating system version: Windows 8.1 Pro, 64-bit.

```
##Time series analysis on R
##Case study "United States"
##yue.liu@wur.nl

install.packages("TTR")
library("TTR")

##Case: USA
##Plz change the file folder to where the data is
Time_Sequence_USA <- read.csv("F:/Thesis/USA_case/Time_Sequence_USA.txt",
header=F)
##convert second to hour
Time_Seq_Hour_USA = rep(0,232)
for (i in 1:232){
  Time_Seq_Hour_USA[i] = sum(Time_Sequence_USA$V2[((i-1)*3600 + 1) :
(i*3600)])
}
##Make a time-series
USA_TS_HOUR =  ts(Time_Seq_Hour_USA, frequency = 24, start = c(5,9))
##STL decomposition result
USA_TS_HOUR_STL = stl(USA_TS_HOUR, 35 , inner = 1, outer = 5)

plot(USA_TS_HOUR_STL, col = "red",main = "Time-series decomposition of
United States")
USA_TS_HOUR_STL_TS = USA_TS_HOUR_STL$time.series
##seasonal adjustment
USA_TS_HOUR_TA = USA_TS_HOUR_STL_TS[,"trend"] +
USA_TS_HOUR_STL_TS[,"remainder"]


plot(USA_TS_HOUR, col = "grey",ylim = c(0,2200),main = "Non-seasonal trends
of United States", ylab = "Tweets per hour", xlab = "Date", lwd = 1)
lines(USA_TS_HOUR_TA, col = "red",lwd = 2)
plot(USA_TS_HOUR_STL_TS[,"seasonal"], main = "Seasonal component of United
States after time-series decomposition",xlab = "time",ylab = "Tweets per
hour", col = "blue")

##lagged difference, with lag = 1(default)
df_USA_TA = diff(USA_TS_HOUR_TA)
plot(df_USA_TA, col = "grey", main = "Lagged differences of non-seasonal
trends of United States", ylab = "Lagged difference of tweets per hour")

##Find patterns of interests, then plot these points on curve
USA_Rem_Events = which(df_USA_TA > 300)
points(5.333333 + USA_Rem_Events/24, df_USA_TA[USA_Rem_Events], col = "red",
type = "p", pch = 1,cex = 1.5,lwd = 2)

plot(USA_TS_HOUR_TA, col = "grey", main = "Non-seasonal trends of United
States with valuable patterns detected",ylab = "Tweets per hour")
points(5.333333 + USA_Rem_Events/24, USA_TS_HOUR_TA[(USA_Rem_Events+1)],
col = "red", type = "p", pch = 1,cex = 1.5,lwd = 2)



## Result in UTC -7 time zone
UTC7_USA_TS_HOUR =  ts(Time_Seq_Hour_USA, frequency = 24, start = c(5,2))
```

```
UTC7_USA_TS_HOUR_STL = stl(UTC7_USA_TS_HOUR, 35 , inner = 1, outer = 5)
plot(UTC7_USA_TS_HOUR_STL, col = "red",main = "Time-series decomposition of
United States")
UTC7_USA_TS_HOUR_STL_TS = UTC7_USA_TS_HOUR_STL$time.series
plot(UTC7_USA_TS_HOUR_STL_TS[,"seasonal"], main = "Seasonal component of
United States after time-series decomposition (UTC -7 Time zone)",xlab =
"time",ylab = "Tweets per hour", col = "blue")
UTC7_seasonal = UTC7_USA_TS_HOUR_STL_TS[,"seasonal"](
UTC7_seasonal_DAY6 = UTC7_USA_TS_HOUR_STL_TS[,"seasonal"][23:46]
##Plot a detailed seasonal component
plot(UTC7_seasonal_DAY6,type = "l", lwd = 2,xaxp = c(0,24, 12),  col =
"blue", main = "Seasonal component of United States in one season (24
hours)", xlab = "Hour in one day (UTC -7 time zone)", ylab = "Tweets per
hour")
abline(v = c(6,12,18))
text(x = 3, y = 300, labels = "Late-night", font = 2)
text(x = 9, y = -500, labels = "Morning", font = 2)
text(x = 15, y = -500, labels = "Afternoon", font = 2)
text(x = 21, y = -500, labels = "Night", font = 2)

##Analyse different patterns, into administrative level: State
##Pattern 1:VA
Time_Sequence_VR<- read.csv("F:/Thesis/VR_case/Time_Sequence_VR.txt",
header=F)
Time_Seq_Hour_VR = rep(0,232)
for (i in 1:232){
  Time_Seq_Hour_VR[i] = sum(Time_Sequence_VR$V2[((i-1)*3600 + 1) :
(i*3600)])
}
VR_TS_HOUR =  ts(Time_Seq_Hour_VR, frequency = 24, start = c(5,9))

VR_TS_HOUR_STL = stl(VR_TS_HOUR, 35 , inner = 1, outer = 5)
plot(VR_TS_HOUR_STL)

VR_TS_HOUR_STL_TS = VR_TS_HOUR_STL$time.series
VR_TS_HOUR_TA = VR_TS_HOUR_STL_TS[,"trend"] +
VR_TS_HOUR_STL_TS[,"remainder"]

plot(VR_TS_HOUR_TA, col = "grey", main = "Non-seasonal trends of
Virginia",ylab = "Tweets per hour")
points(5.333333 + USA_Rem_Events[1]/24,
VR_TS_HOUR_TA[(USA_Rem_Events[1]+1)], col = "red", type = "p", pch = 1,cex
= 1.5,lwd = 2)
text(5.333333 + USA_Rem_Events[1]/24 +0.7,
VR_TS_HOUR_TA[(USA_Rem_Events[1]+1)], "Pattern I")

##Pattern 2:OR
Time_Sequence_OR<- read.csv("F:/Thesis/Oregon_case/Time_Sequence_OR.txt",
header=F)
Time_Seq_Hour_OR = rep(0,232)
for (i in 1:232){
  Time_Seq_Hour_OR[i] = sum(Time_Sequence_OR$V2[((i-1)*3600 + 1) :
(i*3600)])
}
OR_TS_HOUR =  ts(Time_Seq_Hour_OR, frequency = 24, start = c(5,9))

OR_TS_HOUR_STL = stl(OR_TS_HOUR, 35 , inner = 1, outer = 5)
plot(OR_TS_HOUR_STL)

OR_TS_HOUR_STL_TS = OR_TS_HOUR_STL$time.series
```

```
OR_TS_HOUR_TA = OR_TS_HOUR_STL_TS[,"trend"] +
OR_TS_HOUR_STL_TS[,"remainder"]

plot(OR_TS_HOUR_TA, col = "grey", main = "Non-seasonal trends of
Oregon",ylab = "Tweets per hour")
points(5.333333 + USA_Rem_Events[2]/24,
OR_TS_HOUR_TA[(USA_Rem_Events[2]+1)], col = "red", type = "p", pch = 1,cex
= 1.5,lwd = 2)
text(5.333333 + USA_Rem_Events[2]/24 +0.7,
OR_TS_HOUR_TA[(USA_Rem_Events[2]+1)], "Pattern II")

##Pattern 4/5:Kentucky
Time_Sequence_Ken<-
read.csv("F:/Thesis/Kentucky_case/Time_Sequence_Ken.txt", header=F)
Time_Seq_Hour_Ken = rep(0,232)
for (i in 1:232){
  Time_Seq_Hour_Ken[i] = sum(Time_Sequence_Ken$V2[((i-1)*3600 + 1) :
(i*3600)])
}
Ken_TS_HOUR =  ts(Time_Seq_Hour_Ken, frequency = 24, start = c(5,9))

Ken_TS_HOUR_STL = stl(Ken_TS_HOUR, 35 , inner = 1, outer = 5)
plot(Ken_TS_HOUR_STL)

Ken_TS_HOUR_STL_TS = Ken_TS_HOUR_STL$time.series
Ken_TS_HOUR_TA = Ken_TS_HOUR_STL_TS[,"trend"] +
Ken_TS_HOUR_STL_TS[,"remainder"]
#pattern 4
plot(Ken_TS_HOUR_TA, col = "grey", main = "Non-seasonal trends of
Kentucky",ylab = "Tweets per hour")
points(5.333333 + USA_Rem_Events[4]/24,
Ken_TS_HOUR_TA[(USA_Rem_Events[4]+1)], col = "red", type = "p", pch = 1,cex
= 1.5,lwd = 2)
text(5.333333 + USA_Rem_Events[4]/24 +0.7,
Ken_TS_HOUR_TA[(USA_Rem_Events[4]+1)], "Pattern IV")

#pattern 4
plot(Ken_TS_HOUR_TA, col = "grey", main = "Non-seasonal trends of
Kentucky",ylab = "Tweets per hour")
points(5.333333 + USA_Rem_Events[5]/24,
Ken_TS_HOUR_TA[(USA_Rem_Events[5]+1)], col = "red", type = "p", pch = 1,cex
= 1.5,lwd = 2)
text(5.333333 + USA_Rem_Events[5]/24 +0.7,
Ken_TS_HOUR_TA[(USA_Rem_Events[5]+1)], "Pattern V")
```