Article

Local Event Detection Scheme by Analyzing Relevant Documents in Social Networks

Dojin Choi 1, Soobin Park 1, Hunjin Lim 1, Kyoungsoo Bok 2 and Jaesoo Yoo 1,\*

1 Department of Information and Communication Engineering, Chungbuk National University, Chungdae-ro 1, Seowon-Gu, Cheongju, Chungbuk 28644, Korea; mycdj91@chungbuk.ac.kr (D.C.); vivapark530@chungbuk.ac.kr (S.P.); [limhunjin@netsgo.com](mailto:limhunjin@netsgo.com)(H.L.); yjs@chungbuk.ac.kr (J.Y.)

2 Department of SW Convergence Technology, Wonkwang University, Iksandae 460, Iksan, Jeonbuk 54538, Korea; ksbok@wku.ac.kr (K.B.)

**\*** Correspondence: yjs@cbnu.ac.kr; Tel.: +82-43-261-3230

Received: date; Accepted: date; Published: date

**Abstract:** In this paper, we propose a local event detection scheme by analyzing relevant documents in social networks to improve the accuracy of event detection. To detect local events by using geographical data, the proposed scheme embeds them using the geographical data dictionary and generates a weighted keyword graph using social network characteristics. The data left by users in social networks include not only postings but also related documents such as comments and threads. Therefore, the proposed scheme detects the local event based on a keyword graph that is constructed through the analysis of the relevant documents. It can improve the accuracy of local event detection by analyzing relevant documents embedded with region-related information without requiring users to tag geographic data using the geographical data dictionary. In order to verify the superiority of the proposed scheme, we compare it with the existing event detection schemes through various performance evaluations.

**Keywords:** social network service; event detection; relevant documents; keyword graph

1. Introduction

Recently, with the popularization of smart devices, social network services (SNSs) have been widely used to communicate and share information among users [1-3]. SNSs have been used not only to make personal connections but also to rapidly deliver information when a local event occurs. A local event means an event in the real world including state and county fairs, city festivals, circuses, protests, sport games, flea markets, and other public gathering events or naturally occurring things such as disasters. SNS users upload tweets and posts in real-time to share meaningful information when a local event occurs at a particular time and place. When hurricane Sandy hits the eastern part of the U.S in 2012, the refueling fiasco was arisen to secure oil to operate power generators for each household due to lack of electric power supply caused by damaging transmission towers. At that time, people shared conditions, contact numbers, and waiting times of gas stations through SNSs that can share information in real-time. In the situation of a disaster, SNSs have merits that can utilize highly reliable collective intelligence [4].

There have been studied extensively about the event detection schemes based on SNSs. The posts written by users have various information such as hashtags, situations, time, and locations. Text based event detection schemes have been proposed to consider the importance and frequency of keywords by analyzing and extracting meaningful words on hashtags, contents, and comments [5-12]. Text based event detection schemes utilize TF-IDF to extract keywords [5]. Recently, some studies utilize machine learning algorithms such as Word2Vec [9]. Graph based event detection schemes, which they make a graph from extracted keywords and conduct clustering algorithms to find events, have been proposed [13-16]. [13] generated a graph by adapting vector space models. It detects events from clusters that are built based on weights assigned to the graph considering the frequency of co-occurrences.

Recently, local event detection schemes using geo-tag in SNSs have been studied [17-19]. [17] proposed the local event detection scheme that uses time-stamps and geographical information included in the posts. [18] proposed the Geo-tag based local event detection scheme. However, since the proposed scheme in [17] performs text-based event detection, there is a problem with less information that can be obtained in the event of a short posting and with less noise in the process of extracting keywords related to local events, resulting in poor accuracy. In addition, since the actual majority of social network data do not have Geo-tag, the event detection scheme using only the Geo-tag proposed in [8] has a problem with poor accuracy. In this paper, we propose a local event detection scheme by analyzing relevant documents in social networks. The proposed scheme creates a keyword graph by extracting keywords in the social network data. After that, we assign weights to vertices and edges of the keyword graph by considering social network characteristics. We can easily understand relationships between an event and keywords by using the keyword graph. In order to detect local events by using geographical information, the proposed scheme utilizes the geographical dictionary and Geo-tag information to classify the geographical keywords from the keyword graph. The main contributions of this paper are as follows:

1. The proposed scheme uses the geographical dictionary to solve the limitations of the existing schemes that provide and detect local events considering Geo-tags that very sparsely appear with posts. The geographical dictionary means a database that consists of location data mapped to a noun that represents a particular region or area.
2. The proposed scheme analyzes the related documents such as comments and threads to improve accuracy of the event detection scheme. Thereafter, we apply a clustering algorithm to the keyword graph according to weight values.
3. The proposed scheme extracts and provides the local events by merging and dividing clusters by using an edge weight of in-cluster and out-cluster.
4. We compare it with the existing event detection schemes through various performance evaluations in order to verify the superiority of the proposed scheme.

This paper is organized as follows. Section 2 describes related studies. Section 3 describes the proposed scheme’s features and processing methods. In section 4, performance evaluations are described to verify the proposed scheme’s excellence. Finally, section 5 presents the paper’s conclusions and future research.

2. Related Work

Recently, with an activation of the social network services, local event detection schemes have been actively studied through social network service data. Some studies, extracted keywords from posts is utilized for the event detection. [6] proposed not only detecting events but also predicting events in near future through the time analysis by using words related to the detected events. It utilizes the Jacard similarity coefficient to discover event related words. In order to reduce noise about a homomorphic word that are the same form of word but have different meanings, the context of the text containing the word is analyzed by tying the words together in pairs. For example, the word ‘strike’ can have various meanings by the word that comes out with it. When the word occurs with 'baseball', it has a meaning associated with the rules of baseball, and when it is associated with 'lightning', it has a meaning of weather. Many posts in the SNSs have informal style, buzzwords, special characters, and emoticon. Therefore, a text based event detection has always noise problems due to these characteristics.

[13] proposed an event detection scheme by constructing directed keyword graphs such as EventGraph that is created by SNS data to efficiently analyze the correlation of each word. A node in the graph represents a particular word and an edge in the graph represents a relationship between words with a weight that is the number of co-occurrences. It only applies users’ posts to detect local events. Therefore, the accuracy of the local event detection is low since it is hard to determine a particular region by using users’ posts only.

Most events are related to the particular region as well as time. SNSs have been providing Geo-tag function to express their location to other users. [17] proposed a local event detection scheme by utilizing Get-Tag function provided by SNSs. By using the map interface, the map space is divided into squares of equal size and defined as tiles. Each tile has time-spatial information that is extracted from tweets of Twitter. [17] uses STExNMF method [20] to derive keywords and conducts visualizations for detected events. Unfortunately, only about 1% of all tweets have Geo-tag information [21]. Consequently, Geo-tag based local event detection schemes have a low accuracy problem since they only consider 1% data of all contents on SNSs.

Jasmine[22] detects local events in the real-world using geolocation information from microblog documents. The key term extractor in Jasmine extracts key terms that appear three times or more in the Twitter documents. However, it removes all retweets in advance to reduce some noise. The related documents such as comments and threads are very meaningful for detecting a location or describing an event in our scheme.

GeoBurst+[23] detects real-time local events from geo-tagged tweet streams. It proposed a pivot seeking algorithm to generate candidate events. It also proposed a ranking module to classify routine events and local events by using temporal and spatial burstiness with TF-IDF weight. However, as mentioned in Jasmine, Twitter documents have geotagged only 0.7%. It was too few to extract location information.

Eyewitness[24] proposed automatically extracting and summarizing reports of local events from Twitter feeds. It presents a local event detection method by using regression analysis on time series of tweets. Eyewitness used a summarization algorithm for summarizing the detected event such as SumBasic[25]. It eliminates retweets and repeats documents like as Jasmine. They only used geotagged tweets and did not use text content of tweets for event detection.

[26] proposed a two-phase streaming event detection algorithm in Twitter by utilizing Storm with Cassandra. First, it applies the keyword-based algorithm for filtering, and then, it conducts the clustering-based algorithm for event detection. The proposed hybrid method in [26] provides a better balance between accuracy and processing time cost. It focuses on real-time event detection and distributed processing by using keyword burstiness. Therefore, they cannot provide local events.

Firefly[27] detects local news for a given geographical area. In order to overcome the geographical data sparsity, it utilizes users’ profile information and assumes the location of tweets by examining the profile of social friends. In Firefly, the local event detection is very naïve. For example, it defines a threshold, which is the ratio of the number of retweets, to capture local events in the clusters.

[28] proposed a multiscale spatio-temporal real-time event detection approach by exploiting a quad-tree and Poisson variant to dynamically identify events across different spatial scales. DeLLe[29], a methodology for automatically Detecting Latest Local Events from geotagged tweets, detects local events and summarizes them by using a machine learning algorithm such as LSTM. Similar to the [28], the map is divided into grid cells for grouping events. They are difficult to match actual location information since they utilize a quad-tree and grid indexing. Moreover, they only used geotagged social posts of SNS such as Twitter and Flickr.

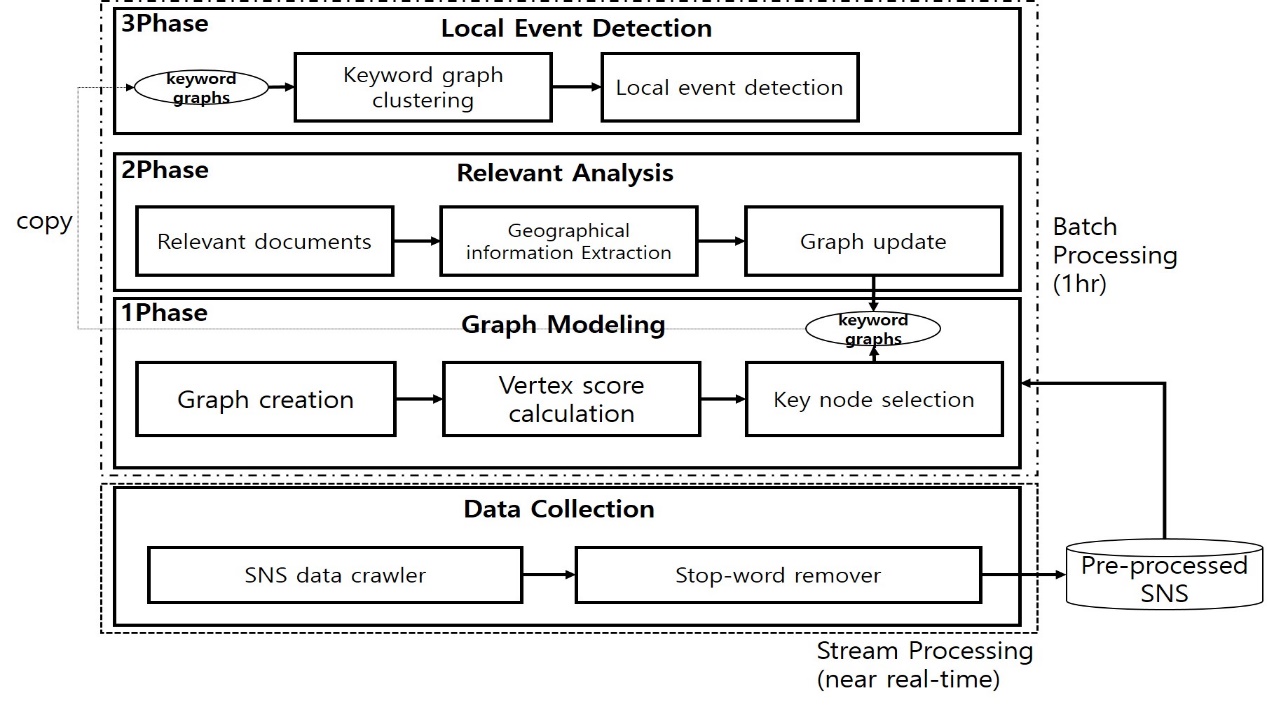
3. The proposed local event detection scheme

3.1. System Architecture

The existing local event detection schemes used text mining techniques and Geo-tags. However, as mentioned above, Geo-tag information appears in tweets very sparsely and thus it is hard to extract geographical information from tweets using text mining techniques. Therefore, it needs to complement additional geographical data. In order to improve local event detection accuracy, we also analyze relevant documents that are associated with previous postings.

In this paper, we propose a local event detection scheme by analyzing relevant documents to solve the problems of the existing schemes such as Geo-tag and text mining based schemes. We embed geographical information to the detected events by using text mining techniques and the geographical dictionary if they do not have Geo-tags. In order to complement sparse geographical data, we exploit relevant documents that have Geo-tags or geographical words. A relevant document means a comment or a thread in SNSs. In Twitter, the message known as ‘tweet’ is originally restricted to 140 characters. Sometimes, users need more than one tweet to express opinions or thinking more and more. In order to overcome the restriction of the tweet, Twitter provides a function, called a thread that is a series of connected tweets from one person. It can be updated and extended consistently. Relevant documents can provide geographical information that does not appear in the previous postings.

Fig. 1 shows an overall architecture of the proposed local event detection scheme. The proposed scheme consists of four modules such as data collection, graph modeling, relevant analysis, and local event detection. The data collection module collects and stores SNS data. It removes stop-words and extracts nouns from the collected data. After the data collection, we perform the local event detection algorithm in sequence through three phases such as graph modeling, relevant analysis, and local event detection. The first phase, the graph modeling module constructs keyword graphs through the pre-processed data and the pre-constructed geographical dictionary in the data collection module. The vertex in the keyword graph represents an extracted keyword and the edge in the keyword graph represents co-occurrence information. We use Geo-tags as much as possible. If users do not provide Geo-tags, we extract regional keywords through text mining algorithms and the geographical dictionary. After that, the graph modeling module assigns keyword type (keyword, regional keyword) to each vertex of the keyword graph. The graph modeling module allocates weights to vertices and edges to calculate vertex scores by using constructed keyword graphs. Finally, the key node selection is performed by utilizing calculated vertex scores. The second phase, the relevant analysis module updates the keyword graph by analyzing relevant documents such as comments and threads that occur within a particular time window that we set 1 hour. The relevant documents are continuously collected from the data collection module. The data collection module also removes stop words from the collected relevant documents. The relevant analysis module extracts geographical information from pre-processed relevant documents. And then, it changes the keyword graph constructed from the graph modeling module by using extracted geographical information. The third phase, the local event detection module detects local events through the clustering algorithm and the measurement of network modularity on an hourly basis. The proposed scheme is targeted only at 'Korean' documents. However, it is an idea that can be applied to any language by properly tuning geographical dictionary and lexical analyzer to suit each language. Buzzword is a relatively common word compared to other words, so it is excluded because it causes noise in event detection. In addition, both emoticons and Hashtag can be analyzed, but only content of tweet was analyzed in the current paper.

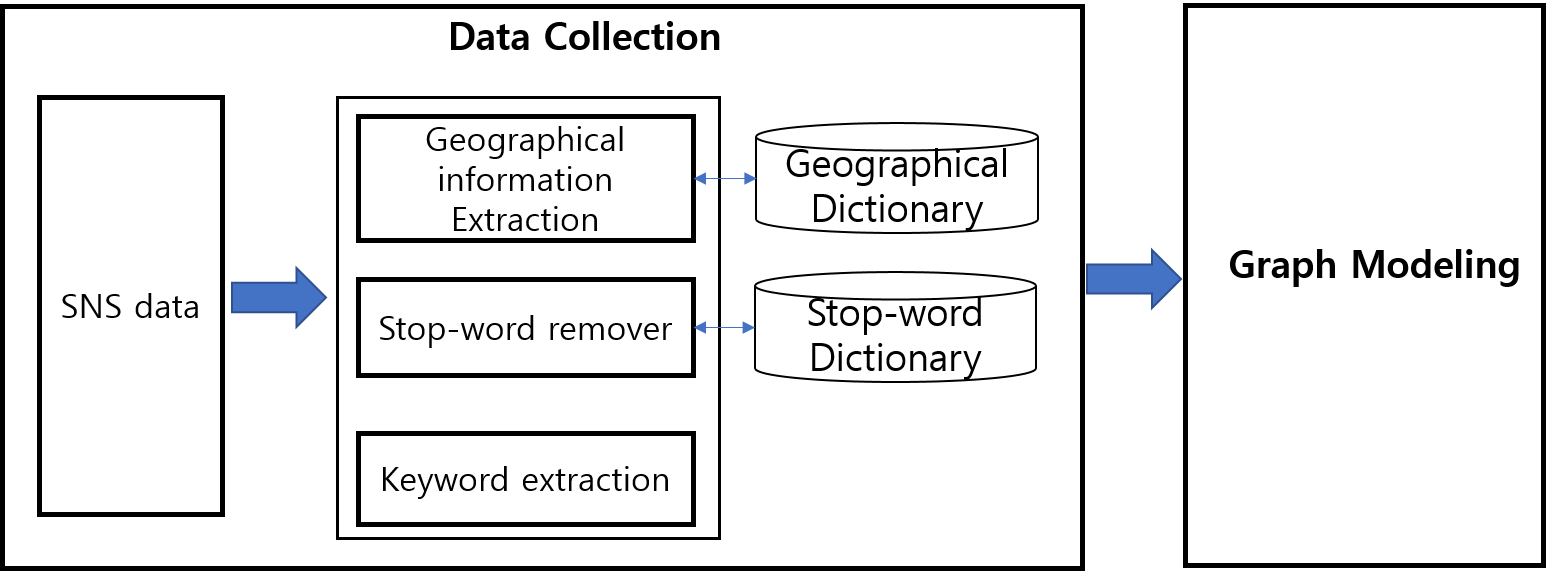


**Figure 1.** Overall architecture of the proposed local event detection scheme.

3.2. Data Collection

In order to detect local events in SNSs, we should collect SNS data generated by users in real-time. The data preprocessing is also required to accurately detect local events since the collected data can contain unnecessary and redundant information such as stop-words, buzzwords, special characters, abbreviations, and emoticons. The data collection not only collects SNS data but also removes redundant and unnecessary data to extract meaningful keywords.

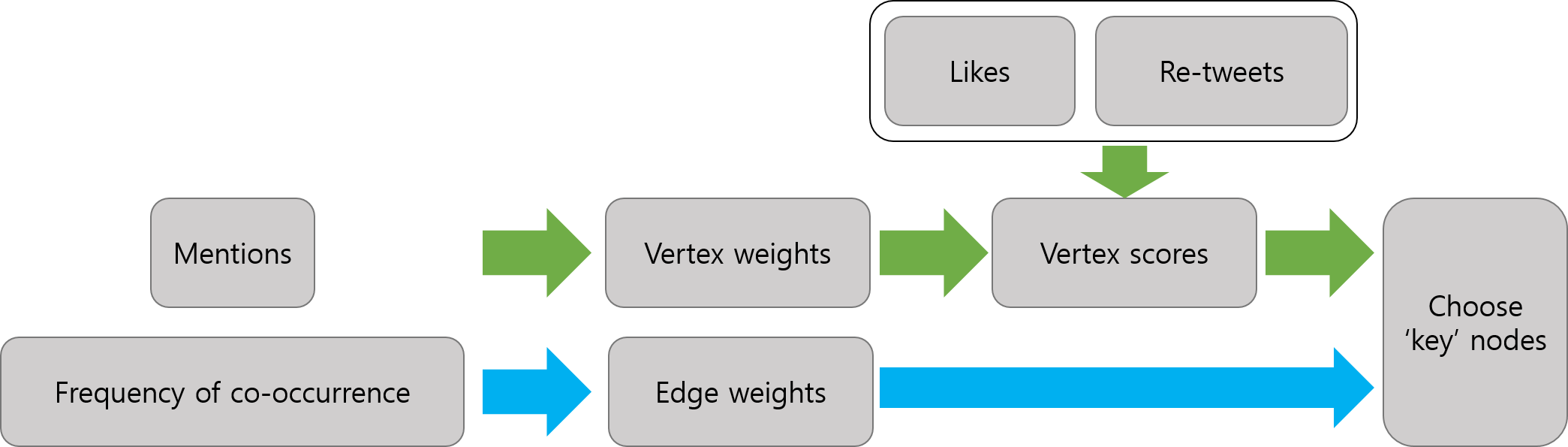
Figure 2 shows the processing procedure of the data collection. It collects SNS data such as posts(tweets), comments, re-tweets, and threads in a particular time window. We extract regional information from the collected data through a geographical dictionary by using Geo-tag information. We assume that the geographical dictionary is pre-constructed. The geographical dictionary is defined according to the administrative district such as village, town, district, county, city, province, metropolitan city, special city, and state. The administrative district has hierarchical features. After that, it gets rid of redundant data from the collected data through the lexical analyzer and the stop-word dictionary. The lexical analyzer splits a sentence into words, and then it removes words that are considered stop-words such as a verb, an adjective, an adverb, a preposition, and etc. We only utilize nouns for the event detection. Lastly, the data collection sends extracted nouns and Geo-tagged data to the module graph modeling for modeling and clustering to extract local events.



**Figure 2.** Processing procedure of the data collection.

3.3. Graph Modeling

We gather keyword sets through the data collection. However, the keyword sets may contain a lot of unnecessary information unrelated to the event. Therefore, we generate a keyword graph considering the word importance and the number of mentions to refine the keyword sets that contain ambiguous information. We easily figure out a keyword correlation and importance through the keyword graph that is assigned a weight value to vertices and edges according to the word importance and the number of mentions. Figure 3 shows the overall procedure of selecting ‘key’ nodes that mean a delegated word in the keyword graph. In order to assign weights, we consider the number of co-occurrences and the number of mentions. After that, to calculate vertex scores, we consider users’ explicit opinions in SNSs such as the number of likes and the number of re-tweets. We use Geo-tag information for choosing regional nodes when the keyword graph is constructed. We embed geographical data using the geographical dictionary if there don’t have Geo-tag data. We construct the keyword graph that is assigned weight values to vertices and edges based on the number of co-occurrences and the number of mentions. Lastly, we perform an initial graph modeling.



**Figure 3.** Overall procedure of selecting ‘key’ nodes

We assign weight values to vertices and edges to use the constructed keyword graph for detecting local events. Through this step, we can easily understand the keyword correlation and the importance according to the SNS characteristics such as likes, re-tweets, and mentions. The frequency of co-occurrences means the keyword correlation between two words. It is utilized for representing the edge weight. For example, ‘cat’ and ‘companion animal’ are very relevant, while ‘cat’ and ‘calendar’ are relatively not associated. Therefore, we should use it for representing this situation. The frequency of co-occurrences is changed according to a particular window size that a user sets up. If the window size is 1, the front and back words of a specific word are contained to the frequency of co-occurrences. The frequency of co-occurrences is normalized into the range [0, 1]. Min-max feature scaling of the edge weight can be formulated by equation (1), where *ewij*, *ewmax*, and *ewmin*denote the frequency of the co-occurrence of vertices *i* and *j*, the highest co-occurrence value, and the lowest co-occurrence value, respectively.

|  |  |
| --- | --- |
| , | (1) |

If we assign the same weight to all vertices that occur within the particular time window *t*, it is hard to understand which keywords are important and related to events. Therefore, we consider users’ explicit opinions and interests to calculate vertex scores since it may contain significant words and meanings. The vertex weight can be formulated by equation (2). The proposed scheme measures vertex weights by utilizing modified TF-IDF(Term Frequency–Inverse Document Frequency). The proposed scheme calculates the term frequency(TF) of keyword *i* like the original method, and the inverse document frequency(IDF) is calculated by the ratio of the IDF of the current time window(*t*) to the IDF of the previous time window(*t-1*).

|  |  |
| --- | --- |
| , | (2) |

We measure vertex scores by using vertex weights and users’ explicit opinions such as like and re-tweets. We can use explicit opinions as weights since posts that contain a lot of explicit opinions from other users may be more reliable and important. The vertex score can be formulated by equation (3), where and denote the number of likes of keyword *i* and the number of re-tweets of the keyword within the particular time window *t*, respectively. We apply the logarithm to the sum of likes and re-tweets to scale it.

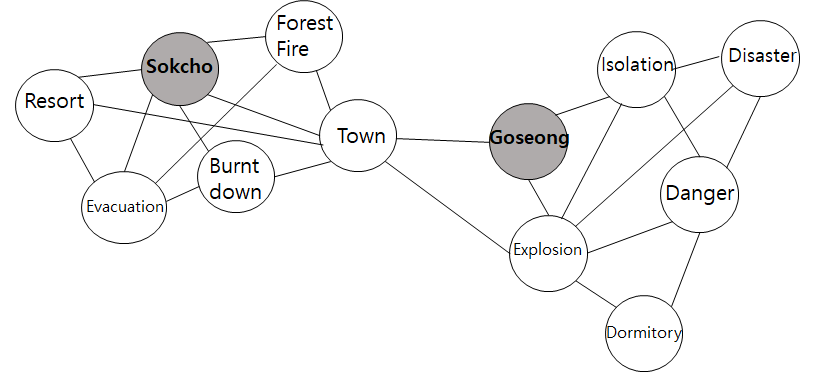
|  |  |
| --- | --- |
| , | (3) |

In order to reveal ‘key’ nodes, we measure Text-Rank [30] scores on the weighted keyword graph. Text-Rank is frequently used for extracting primary keywords from the entire graph. We perform clustering based on ‘key’ nodes that are selected according to the Text-Rank algorithm. The Text-Rank score can be formulated by equation (4), where *vit*, *Ni*, and *d* denote the particular vertex *i* over time window *t*, neighbors of vertex *i* in the weighted keyword graph, and the damping factor, respectively. We initialize *tr(vit)* value with *vsit* that is the vertex score calculated by equation (3). The factor *d* that adjusts the random probability variable is usually set to 0.85 [31].

|  |  |
| --- | --- |
| , | (4) |

The graph clustering based on ‘key’ nodes makes it easier to detect words associated with the event and improves the performance than the graph clustering based on entire keyword vertices. We choose top-k ‘key’ nodes according to the Text-Rank scores by the descending order. Geo-tags or regional nodes that are built by the geographical dictionary are also chosen as ‘key’ nodes since they have the important role in detecting local events.

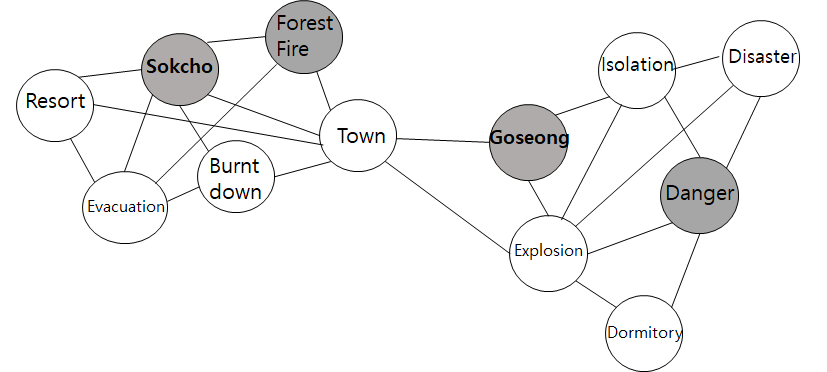
Figure 4 shows the procedure of constructing the keyword graph and choosing ‘key’ nodes. We construct the initial keyword graph based on pre-processed data as shown in Figure 4(a). ‘Sokcho’ and ‘Goseong’ are labeled as regional nodes. We calculate and assign vertex and edge weights by using equations (1) and (2). After that, we perform Text-Rank on the graph. First, we compute vertex scores by using equation (3). Second, we assign a vertex score to each vertex of the original graph. Finally, we can measure Text-Rank scores by using equation (4). We sort the Text-Rank scores by descending order and select top-k(k is 4 in this example) nodes(‘Forest Fire’, ‘Sokcho’, ‘Goseong’, and ‘Danger’) as shown in Figure 4(b). Finally, we can get the graph with the ‘key’ nodes as shown in Figure 4(c). A graph is generated based on tweets that occur within a given unit of time window (1 hour). To distinguish events, since graph clustering is performed on a key node basis through Text-Rank based 'Key' node selection, each cluster represents one event. Therefore, related words are clustered among themselves, allowing them to distinguish between local events.



(**a**)



(**b**)



(**c**)

**Figure 4.** Overall procedure constructing the keyword graph and choosing ‘key’ nodes: (a) An initial keyword graph with labeled regional nodes; (b) Text-Rank scores each keyword; (c) ‘key’ nodes assigned to the keyword graph.

3.4. Relevant Document Analysis

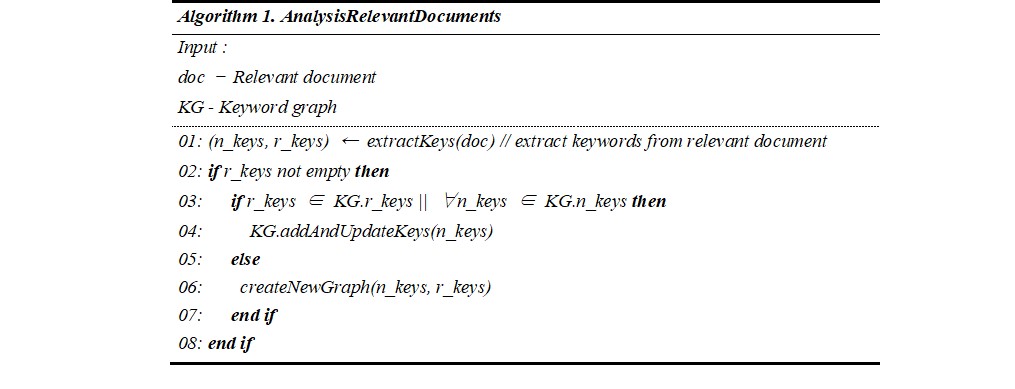
It needs to consider analyzing relevant documents for detecting events based on SNS data. SNS users can add or put relevant documents to their own postings and other users’ postings. Users upload related documents such as comments and threads to express their opinions. Especially, documents with geographic information that is very useful for detecting local events are used in order to increase detection accuracy. As mentioned above, posts containing many explicit opinions may have important content or keywords. Therefore, in this paper, we perform local event detection through relevant documents analysis. We can also easily update the graph because relevant documents already express relationships among postings.

Figure 5 shows an example of relevant documents. Although a tweet of user 1 includes event information, it is difficult to determine a particular region. Region information is added through user1’s answer to user2’s comment. The existing schemes do not consider relevant documents, while the proposed scheme considers additional information through analyzing relevant documents for updating the keyword graph.



**Figure 5.** Example of relevant documents.

Figure 6 shows the relevant document analysis algorithm. The input parameters are doc(the relevant document) and KG(the keyword graph). We extract keywords (normal keywords(n\_keys), regional keywords(r\_keys)) from the relevant document(Line 1). We use only documents with regional keywords to reduce the complexity of the analysis module(Line 2). If KG contains the extracted regional keywords(r\_keys) or all n\_keys are included in KG, we add n\_keys to the keyword graph(KG) and update weights and vertex scores(Line 3-4). Otherwise, we create a new keyword graph by using n\_keys and r\_keys since it is likely to be a newly detected event(Line 5-6).



**Figure 6.** Relevant document analysis algorithm.

3.5. Local Event Detection

The proposed scheme performs a clustering algorithm based on edge weights in the keyword graph to classify local events. An edge weight represents the keyword correlation coefficient that means the higher the weight is, the more frequently co-occurring they are. The clustering algorithm combines the ‘key’ nodes with the (one-hop) neighbors that have a higher edge weight than a threshold (α). The threshold is selected based on the network modularity that embodies the strength of the division of a network into modules(clusters). The network modularity is often used in optimization methods for detecting communities(clusters) in a graph. We use it for detecting and grouping events to efficiently find local events.

The network modularity is formulated by equation (5), where *m*, *n*, *ewij* and *ki* denote the number of edges in the graph, the number of vertices in the graph, the edge weight between vertex *i* and *j* that is calculated by equation (1), and the sum of edge weights of vertex *i*, respectively. is a boolean function that returns 1 or 0 depending on whether vertex *i* and vertex *j* are in the same cluster.

|  |  |
| --- | --- |
| , | (5) |

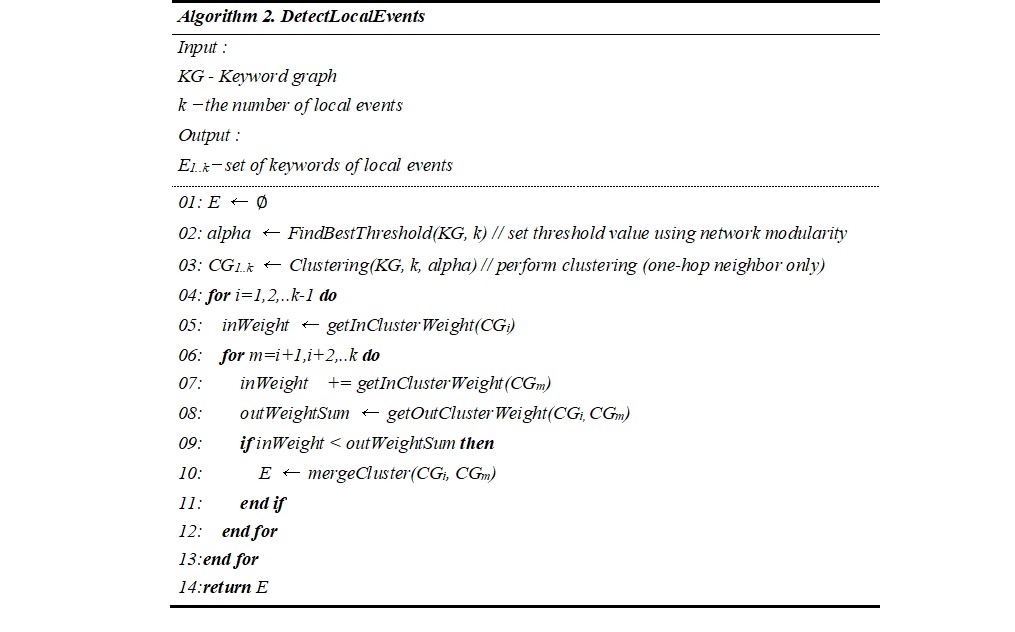
The procedure of merging and splitting clusters is needed for providing “k” events to users. The proposed scheme provides “k” clusters to users as the local event by using the in-cluster and the out-cluster weight.

Figure 7 shows an example of detecting local events of the proposed scheme. First, we just make clusters according to the above mentioned procedure. After that, we can get three clusters as shown in figure 6. The in-cluster weight is calculated based on the average of edge weights in the cluster. The out-cluster weight is calculated based on the sum of edge weights of inter-cluster. In this example, the in-cluster weights of *C1* and *C2* are 0.66 and 0.53, respectively. The out-cluster weight between *C1* and *C2* is 1.55. We compare the sum of the in-cluster weights with the out-cluster weight. In this case, the out-cluster weight is higher than the sum of the in-cluster weights (0.66+0.53=1.19 < 1.55). Therefore, we merge *C1* and *C2* since *C1* (or *C2*) is closely related with *C2* (or *C1*). Likewise, we compare the sum of the in-cluster weights with the out-cluster weight between *C1* and *C3*. The out-cluster weight is lower than the sum of the in-cluster weights (in. 0.66+0.74 > out. 0.40+0.32). We just remove the connected edges of *C1* and *C3.* Finally, we provide users with detected(clustered) local events as shown in figure 6(b).

|  |  |
| --- | --- |
| EMB00003fd4362c  (a) | EMB00003fd4362c  (b) |

**Figure 7.** Example of detecting local events: (a) clustering; (b) detecting local events by merging and splitting clusters.

Figure 8 shows the local event detection algorithm. The input parameters are KG(the keyword graph) and k(the number of local events). The return values are the keyword sets of the detected local events. We set the threshold alpha and the output data to initial values (Line 1-2). We use the network modularity to find the best threshold value by using equation (5). As already mentioned, the threshold is used for the “k” clustering algorithm that combines one-hop neighbors, which they have a higher edge weight than the threshold that is induced by the network modularity, with the ‘key’ node(Line 3). Finally, we merge clusters “a” and “b” iteratively by comparing the sum of the in-cluster weights with the out-cluster weight(Line 4-13). The merged cluster is inserted to the result E(Line 10). We return the result E as a set of the detected local events.



**Figure 8.** Local event detection algorithm.

4. Experiments

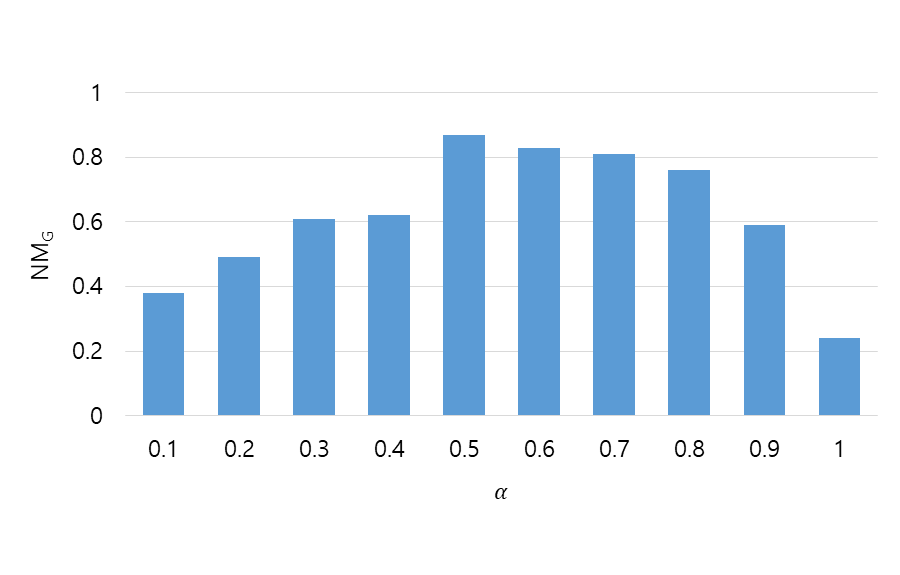
We verify the superiority of the proposed scheme by comparing its performances with the existing event detection schemes. The experimental environment is shown in Table 1. We conducted various experiments on single server environments, where the server was equipped with an Intel core i5-3570 CPU 3.40GHz and 16GB of memory. We implemented the proposed scheme by using Python 2.7 on Window OS. We collected 106,095 tweets and 147,326 relevant documents by using the Twitter Scraper API. We collect 1 month of tweets and documents (from April 1, 2019). In our paper, the ground truth was annotated by human users. In the experiments, we used 50 local events as the ground truth. The ground truth is called the true set in the paper. The recall was calculated by changing K from 5 to 20 (maximum/10 to max/2.5). Recall calculation values are rounded. Accuracy is calculated by giving 0.5 if the clustered event is the same as the true set, and 0.5 if the keywords that can be judged to be the same meaning as the tagged event category are clustered.

**Table 1.** The experimental environment.

|  |  |
| --- | --- |
| **Name** | **Value** |
| Processor | Intel i5-3570k, 3.4GHz, 4 Core |
| Memory | 16GB |
| # of tweets | 106,095 |
| # of relevant documents | 147,326 |

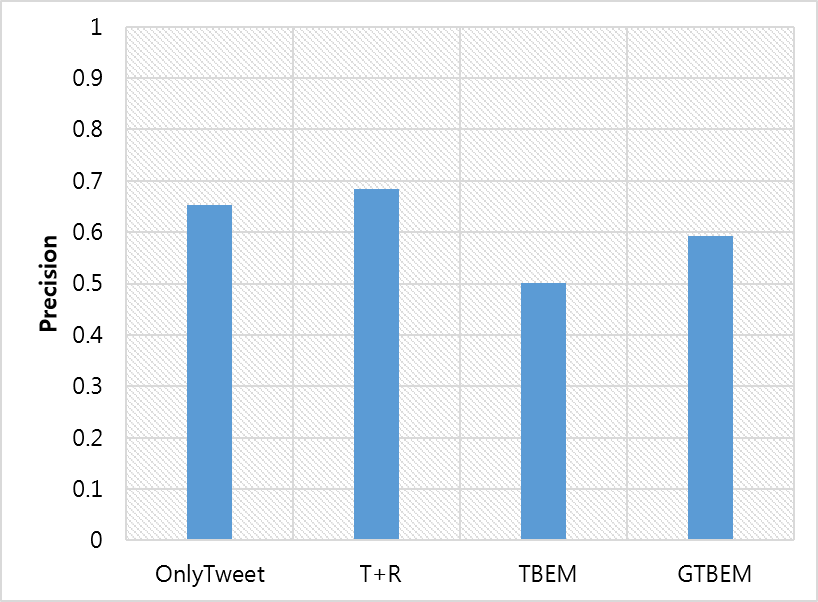
The existing schemes such as text based event detection[6] and Geo-tag based event detection[17] were chosen for comparison. We compare Precision, Recall, and F-measure to show the excellence of the proposed scheme. We call the event detection scheme does not consider relevant documents OnlyTweet. The proposed scheme, text based event detection[6], and Geo-tag based event detection[17] are called T+R, TBEM, and GTBEM, respectively.

The results of the clustering depend on a threshold. Therefore, we derive an optimized value through experiments. Figure 9 shows a comparison of NMG according to different thresholds. In experiments, thresholds were assigned from 0.1 to 1. When the threshold is 0.1, we cannot extract important local events since the redundant keywords were contained in the event cluster. Since the threshold shows the best NMG over the other NMG when the threshold is 0.5, we set the threshold to 0.5.



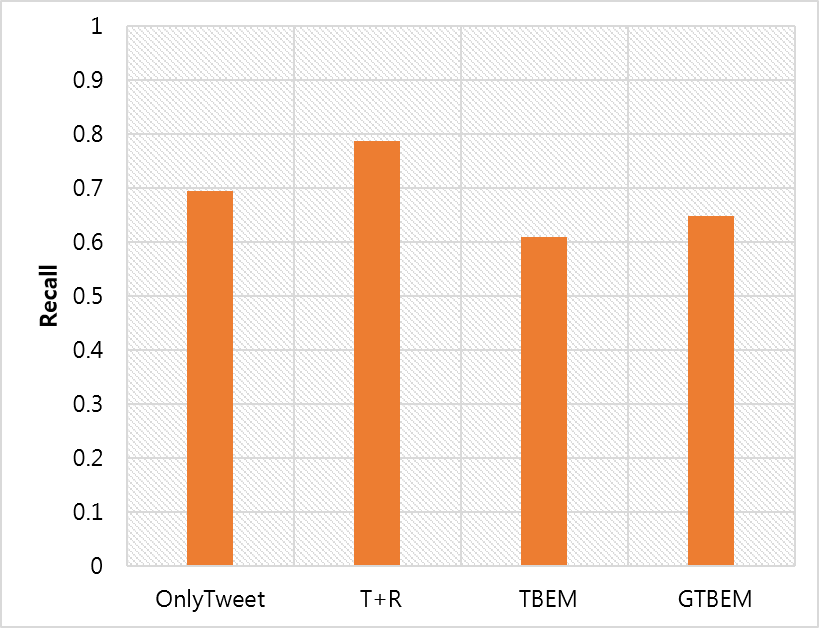
**Figure 9.** Comparison of NMG (Network modularity) according to different thresholds.

Figure 10 shows a comparison of precision according to the local event detection schemes. TBEM showed relatively low performance (about 50%) since it is hard to derive regional information since it uses only a text-mining algorithm. About 50% precision means that the detected local events are only half of ground truth. On the other word, the precision of TBEM was the lowest. GTBEM showed better performance than TBEM (about 59%). However, it only considered Geo-tags. OnlyTweet that does not consider relevant documents ranked the second position (about 65%). It means that Tweets have contained more regional information than Geo-tagged Tweets. The proposed scheme showed the best performance in terms of precision. As a result, the proposed scheme improved performance by about 4% through the relevant document analysis compared to the OnlyTweet.



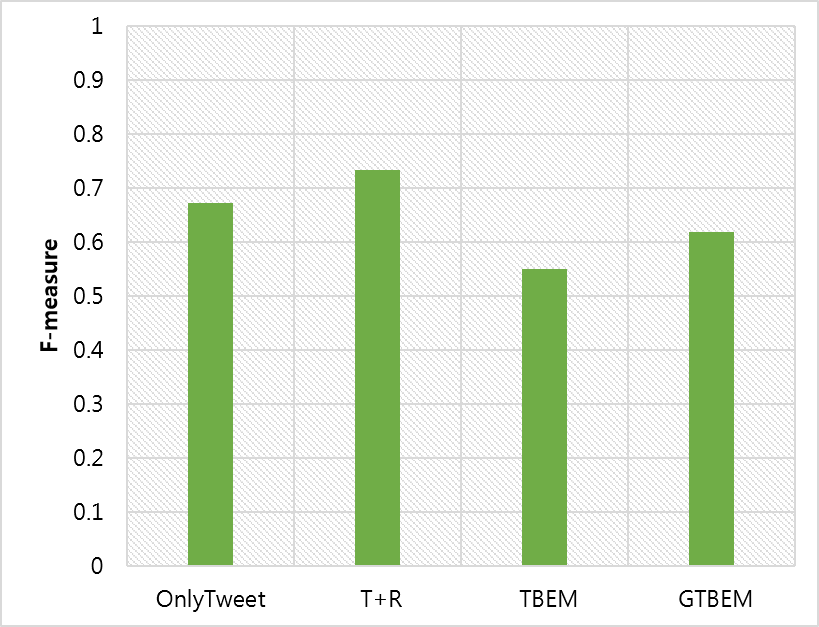
**Figure 10.** Comparison of precision according to the local event detection schemes.

Figure 11 shows a comparison of recall according to the local event detection schemes. Like the experiment results of precision, our scheme achieves the best performance over the existing schemes by improving performance about 16%. The recall performance of TBEM, GTBEM, OnlyTweet, and the proposed scheme achieved about 62%, 65%, 69%, and 78%, respectively. As shown above, the recall of TBEM is also much lower than the others, since it is hard to extract regional information. The recall performance of GTBEM achieved 60% or more, since it takes into account only Geo-Tag information which means detected events with at least 1 or more Geo-Tag information. The proposed scheme achieved the highest recall of about 0.8(80%). It means that relevant local events were detected well since we utilize not only the geographical dictionary but also relevant documents to complement spare geographical data.



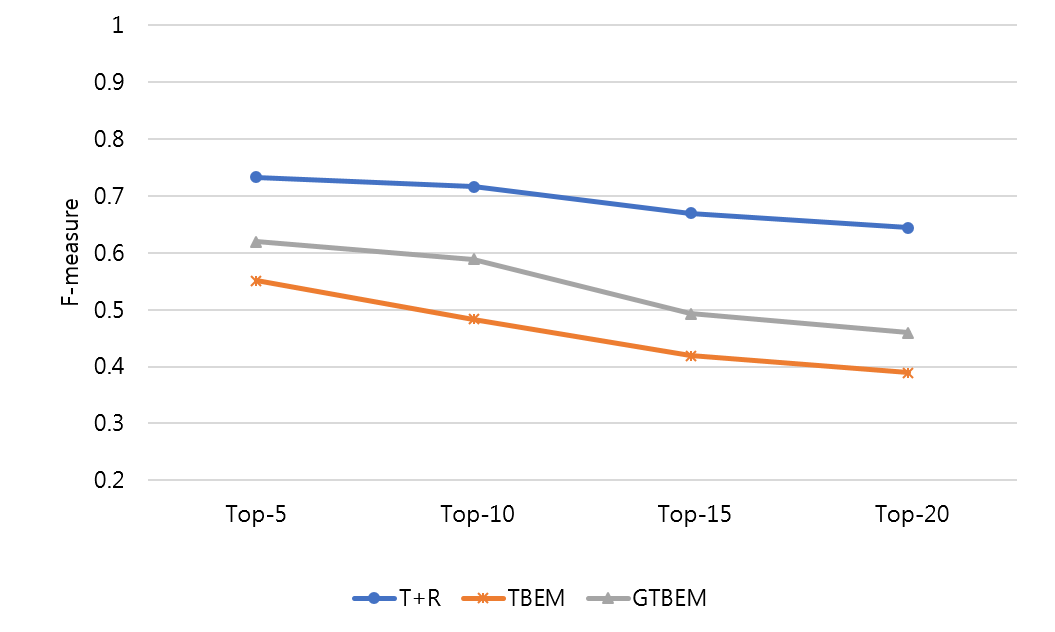
**Figure 11.** Comparison of recall according to the local event detection schemes.

Figure 12 shows a comparison of F-measure according to the local event detection schemes. F-measure is the harmonic mean of precision and recall. Therefore, the performance results of F-measure are reasonable since the proposed scheme already achieved the highest precision and recall over the existing schemes as shown in figures 10 and 11. The F-measures of TBEM, GTBEM, OnlyTweet, and the proposed scheme achieved about 55%, 63%, 67%, and 73%, respectively. As a result, the proposed scheme achieves higher performance than the existing schemes about average 11% or more since it considers both the relevant document analysis based on text mining and Geo-tag.



**Figure 12.** Comparison of F-measure according to the local event detection schemes.

Figure 13 shows F-measure according to the number of top-k events. When k is 5, all schemes have a high F-measure. The F-measure of TBEM, GTBEM, and the proposed scheme achieved about 55%, 62%, and 74%, respectively. The proposed scheme achieves higher performance than the existing schemes about average 15% or more when k is 5. The F-measures of all of the schemes decrease as k increases since precision decreases as the number of events to be detected increases. Since GTBEM cannot detect non-regional events, performance results are very low as k increases. Although TBEM and GTBEM can provide meaningful top-20 events, most of the results did not involve “local” events since they are hard to extract regional information precisely. As a result, the proposed scheme significantly improves the performance of the local event detection in terms of precision, recall, and F-measure.



**Figure 13.** F-measure according to the number of top-k events.

5. Conclusions

We have proposed an efficient local event detection scheme by analyzing relevant documents in a social network service such as Twitter. The proposed scheme utilizes the geographical dictionary to supplement non-Geo-Tagged postings in order to detect local events by using geographical information. The proposed scheme has analyzed the related documents such as comments and threads to improve the accuracy of detecting local events. The proposed scheme used a great deal of non-Geo-Tagged data on SNS to detect local events, and geographical dictionary was used to extract detailed local information. In addition, a modified TF-IDF with time characteristics was proposed to detect the time of event occurrence and keyword burstiness. Cluster weights were proposed to identify events and remove unnecessary keywords. The proposed cluster weight allows the user to generate the desired k events. It was shown through various performance evaluations the proposed scheme significantly improved performances in terms of precision, recall, and F-measure compared to the existing event detection schemes. In the future, we will secure various clean data sets and conduct additional experiments. In addition, we are going to expand our research into local event detection to distinguish between ongoing and resolved events, and to come up with an aggregation strategy for events spreading over multiple hours.

**Author Contributions:** Conceptualization, D.C., S.P., K.B., H.L., and J.Y.; methodology, D.C., S.P., K.B., and J.Y.; validation, D.C., S.P., and K.B.; formal analysis, D.C., S.P., H.L., and K.B.; writing—original draft preparation, , D.C., S.P., H.L., and K.B.; writing—review and editing, J.Y.

**Acknowledgments:** This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT). (No. 2019R1A2C2084257), by Next-Generation Information Computing Development Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT (No. NRF-2017M3C4A7069432), and by Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT)(No.B0101-15-0266, Development of High Performance Visual BigData Discovery Platform for Large-Scale Realtime Data Analysis) Large-Scale Realtime Data Analysis.

**Conflicts of Interest:** The authors declare no conflict of interest.

References

1. Hall, J.A. When is Social Media use Social Interaction? Defining Mediated Social Interaction. New Media & Society **2018***, 20*, 162-179.
2. Oh, S.; Syn, S.Y. Motivations for Sharing Information and Social Support in Social Media: A Comparative Analysis of F Acebook, T Witter, D Elicious, Y Ou T Ube, and F Lickr. Journal of the Association for Information Science and Technology **2015***, 66*, 2045-2060.
3. Chang, S.E.; Liu, A.Y.; Shen, W.C. User Trust in Social Networking Services: A Comparison of Facebook and LinkedIn. Comput. Hum. Behav. **2017***, 69*, 207-217.
4. Kryvasheyeu, Y.; Chen, H.; Moro, E.; Van Hentenryck, P.; Cebrian, M. Performance of Social Network Sensors during Hurricane Sandy. PLoS One **2015***, 10*, e0117288.
5. Zhu, Z.; Liang, J.; Li, D.; Yu, H.; Liu, G. Hot Topic Detection Based on a Refined TF-IDF Algorithm. IEEE Access **2019***, 7*, 26996-27007.
6. Hossny, A.H.; Mitchell, L. Event Detection in Twitter: A Keyword Volume Approach. In 2018 IEEE International Conference on Data Mining Workshops (ICDMW); pp. 1200-1208.
7. Cui, A.; Zhang, M.; Liu, Y.; Ma, S.; Zhang, K. Discover Breaking Events with Popular Hashtags in Twitter. In Proceedings of the 21st ACM International Conference on Information and Knowledge Management; pp. 1794-1798.
8. Ardon, S.; Bagchi, A.; Mahanti, A.; Ruhela, A.; Seth, A.; Tripathy, R.M.; Triukose, S. Spatio-Temporal and Events Based Analysis of Topic Popularity in Twitter. In Proceedings of the 22nd ACM International Conference on Information & Knowledge Management; pp. 219-228.
9. Cui, W.; Wang, P.; Du, Y.; Chen, X.; Guo, D.; Li, J.; Zhou, Y. An Algorithm for Event Detection Based on Social Media Data. Neurocomputing **2017***, 254*, 53-58.
10. Gao, W.; Fang, Y.; Wang, Y.; Zhang, F. HRCE: Detecting Food Security Events in Social Media. In Journal of Physics: Conference Series; pp. 012090.
11. Iyda, J.J.; Geetha, P. Keyword-Based Approach for Detecting Civil Unrest Events from Social Media. In EAI International Conference on Big Data Innovation for Sustainable Cognitive Computing; pp. 287-298.
12. Geetha, R.; Devi, P.S.; Karthika, S. # BiggBoss—Long-Run Event Detection and Sentiment Mining in Twitter. In *Information and Communication Technology for Sustainable Development*.; Anonymous .; Springer, 2020, pp. 229-237.
13. He, J.; Liu, Y.; Jia, Y. EventGraph Based Events Detection in Social Media. In International Conference of Pioneering Computer Scientists, Engineers and Educators; pp. 150-160.
14. Genc, H.; Yilmaz, B. Text-Based Event Detection: Deciphering Date Information using Graph Embeddings. In International Conference on Big Data Analytics and Knowledge Discovery; pp. 266-278.
15. Liu, Y.; Peng, H.; Li, J.; Song, Y.; Li, X. Event Detection and Evolution in Multi-Lingual Social Streams. Frontiers of Computer Science **2020***, 14*, 1-15.
16. Pagare, R.; Khare, A.; Chaudhary, S. (T-ToCODE): A Framework for Trendy Topic Detection and Community Detection for Information Diffusion in Social Network. In *Data Management, Analytics and Innovation*.; Anonymous .; Springer, 2020, pp. 613-632.
17. Zhang, C.; Liu, L.; Lei, D.; Yuan, Q.; Zhuang, H.; Hanratty, T.; Han, J. Triovecevent: Embedding-Based Online Local Event Detection in Geo-Tagged Tweet Streams. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; pp. 595-604.
18. Choi, M.; Shin, S.; Choi, J.; Langevin, S.; Bethune, C.; Horne, P.; Kronenfeld, N.; Kannan, R.; Drake, B.; Park, H. Topicontiles: Tile-Based Spatio-Temporal Event Analytics Via Exclusive Topic Modeling on Social Media. In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems; pp. 1-11.
19. Zhang, S.; Cheng, Y.; Ke, D. Event-Radar: Real-Time Local Event Detection System for Geo-Tagged Tweet Streams. arXiv preprint arXiv:1708.05878 **2017**.
20. Shin, S.; Choi, M.; Choi, J.; Langevin, S.; Bethune, C.; Horne, P.; Kronenfeld, N.; Kannan, R.; Drake, B.; Park, H. Stexnmf: Spatio-Temporally Exclusive Topic Discovery for Anomalous Event Detection. In 2017 IEEE International Conference on Data Mining (ICDM); pp. 435-444.
21. Abdelhaq, H.; Sengstock, C.; Gertz, M. Eventweet: Online Localized Event Detection from Twitter. Proceedings of the VLDB Endowment **2013***, 6*, 1326-1329.
22. Watanabe, K.; Ochi, M.; Okabe, M.; Onai, R. Jasmine: A Real-Time Local-Event Detection System Based on Geolocation Information Propagated to Microblogs. In Proceedings of the 20th ACM International Conference on Information and Knowledge Management; pp. 2541-2544.
23. Zhang, C.; Lei, D.; Yuan, Q.; Zhuang, H.; Kaplan, L.; Wang, S.; Han, J. GeoBurst Effective and Real-Time Local Event Detection in Geo-Tagged Tweet Streams. ACM Transactions on Intelligent Systems and Technology (TIST) **2018**, 9, 1-24.
24. Krumm, J.; Horvitz, E. Eyewitness: Identifying Local Events Via Space-Time Signals in Twitter Feeds. In Proceedings of the 23rd Sigspatial International Conference on Advances in Geographic Information Systems; pp. 1-10.
25. Vanderwende, L.; Suzuki, H.; Brockett, C.; Nenkova, A. Beyond SumBasic: Task-Focused Summarization with Sentence Simplification and Lexical Expansion. Information Processing & Management **2007**, 43, 1606-1618.
26. Sahin, O.C.; Karagoz, P.; Tatbul, N. Streaming Event Detection in Microblogs: Balancing Accuracy and Performance. In International Conference on Web Engineering; pp. 123-138.
27. Wei, H.; Sankaranarayanan, J.; Samet, H. Enhancing Local Live Tweet Stream to Detect News. GeoInformatica **2020**, 1-31.
28. George, Y.; Karunasekera, S.; Harwood, A.; Li, K.H. Spatio-Temporal Event Detection using Poisson Model and Quad-Tree on Geotagged Social Media. In 2019 IEEE International Conference on Big Data (Big Data); pp. 2247-2256.
29. Wei, H.; Zhou, H.; Sankaranarayanan, J.; Sengupta, S.; Samet, H. DeLLe: Detecting Latest Local Events from Geotagged Tweets. In Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Analytics for Local Events and News; pp. 1-10.
30. Mihalcea, R.; Tarau, P. Textrank: Bringing Order into Text. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing; pp. 404-411.
31. Brin, S.; Page, L. The Anatomy of a Large-Scale Hypertextual Web Search Engine. **1998**.

|  |  |
| --- | --- |
| copyRight | © 2020 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/). |