applied sciences

MDPI

Article

Local Event Detection Scheme by Analyzing Relevant Documents in Social Networks

Dojin Choi [1], Soobin Park [1], Dongho Ham [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.); ii8858@naver.com (D.H.); limhunjin@netsgo.com (H.L.)

[2] Department of SW Convergence Technology, Wonkwang University, Iksandae 460, Iksan, Jeonbuk 54538, Korea; ksbok@wku.ac.kr

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

Featured Application: Our local event detection scheme can be used in various applications such as traffic flow control services, event location finding services, intrusion detection services, and disaster prevention services. In these applications, our scheme can be used to find the local events in the real world, including accidents, state and county fairs, city festivals, circuses, protests, sport games, flea markets, other public gathering events, and natural disasters.

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 a 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. In this way, the proposed scheme detects a local event based on a keyword graph that is constructed through the analysis of the relevant documents. This can improve the accuracy of local event detection by analyzing relevant documents embedded with region-related information using a geographical data dictionary, without requiring users to tag geographic data. 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

check for updates

Citation: Choi, D.; Park, S.; Ham, D.; Lim, H.; Bok, K.; Yoo, J. Local Event Detection Scheme by Analyzing Relevant Documents in Social Networks. *Appl. Sci.* **2021**, *11*, 577. https://doi.org/10.3390/app11020577

Received: 16 December 2020
Accepted: 6 January 2021
Published: 8 January 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

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 incidents, such as disasters. SNS users upload information and posts in real time to share meaningful information when a local event occurs at a particular time and place. When Hurricane Sandy hit the eastern part of the U.S. in 2012, a refueling fiasco arose to secure oil in order to operate power generators for each household due to the lack of electric power supply caused by damaged transmission towers. At that time, people shared conditions, contact numbers, and waiting times of gas stations through SNSs that allow users to share information in real time. In the situation of a disaster, SNSs have merits that can allow individuals to utilize highly reliable collective intelligence [4].

Event detection schemes based on SNSs have been studied extensively. The posts written by users have various information, such as hashtags, situations, time, and locations.

Text-based event detection schemes have been proposed for considering 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 term frequency–inverse document frequency (TF-IDF) to extract keywords [5]. Recently, some studies utilized machine learning algorithms such as Word2Vec [9]. Graph-based event detection schemes, which involve making a graph from extracted keywords and conducting clustering algorithms to find events, have also been proposed [13–16]. The authors in [13] generated a graph by adapting vector space models; the graph can detect 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-tags in SNSs have been studied [17–19]. Zhang et al. [17] proposed a local event detection scheme that uses time-stamps and geographical information included in the posts. Choi et al. [18] proposed a geo-tag based local event detection scheme. However, since the proposed scheme in [17] performs text-based event detection, there is a problem when little information is obtained in the event of a short posting and hence 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-tagging, the event detection scheme using only the geo-tagging approach 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 the 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 a 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 a geographical dictionary to solve the limitations of the existing schemes that provide and detect local events, considering that geo-tags appear very sparsely with posts. The geographical dictionary refers to 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 the accuracy of the event detection scheme. After the analysis, 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 the proposed scheme 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. In some studies, extracting keywords from posts is utilized for event detection. The authors in [6] proposed not only detecting events but also predicting events in the near future through a 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 the noise generated by a homomorphic word that is the same form of another word but both 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 depending on the words that comes along with it. When "strike" occurs with "baseball", it has a

meaning associated with the rules of baseball, and when it is associated with "lightning", it has a meaning associated with weather. Many posts in the SNSs have an informal style, buzzwords, special characters, and emoticons. Therefore, a text-based event detection always has noise problems due to these characteristics.

The authors in [13] proposed an event detection scheme by constructing directed keyword graphs (such as EventGraph) that are 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 determined by 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. Most SNSs provide a geo-tagging function to users so that they can express their location to other users. The authors in [17] proposed a local event detection scheme by utilizing the geo-tagging function provided by SNSs. Using a 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 (i.e., posted messages) of Twitter. The authors in [17] used the spatio-temporally exclusive topic discovery based on nonnegative matrix factorization (STExNMF) method [20] to derive keywords and to conduct 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.

The Jasmine system in [22] can detect 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 Twitter documents. However, it removes all retweets (i.e., tweets reposted by other users) in advance to reduce some noise. The related documents such as comments and threads are very meaningful for detecting a location or for describing an event in our scheme.

GeoBurst+ in [23] detects real-time local events from geo-tagged tweet streams. The authors proposed a pivot seeking algorithm to generate candidate events. They 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 the study for Jasmine, only 0.7% of Twitter documents are geo-tagged, which is far too few for extracting location information.

Eyewitness in [24] was proposed for automatically extracting and summarizing reports of local events from Twitter feeds. The authors presented a local event detection method by using a regression analysis on the time series of tweets. Eyewitness used a summarization algorithm for summarizing a detected event, such as the SumBasic algorithm [25]. It eliminates retweets and repeated documents, similar to Jasmine. The system only used geo-tagged tweets and did not use text content of tweets for event detection.

The authors in [26] proposed a two-phase streaming event detection algorithm in Twitter by utilizing Storm with Cassandra. First, the system applies a keyword-based algorithm for filtering, and then it conducts a 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. For this reason, the method cannot provide local events.

Firefly in [27] detects local news using 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.

George et al. [28] proposed a multiscale spatiotemporal real-time event detection approach by exploiting a quad-tree and Poisson variant to dynamically identify events across different spatial scales. detecting latest local events (DeLLe) [29], a methodology for

automatically detecting the latest local events from geo-tagged tweets, detects local events and summarizes them by using a machine learning algorithm such as long short-term memory (LSTM). Similar to [28], it uses a map that is divided into grid cells for grouping events. It is difficult to use this for matching actual location information since it utilizes a quad-tree and grid indexing. Moreover, the method only used geo-tagged social posts of SNS such as Twitter and Flickr.

Amato et al. [30] proposed a real-time disaster detection and an alert system based on multimedia big data. The authors exploited an a priori algorithm to classify and track events. They also presented an influence diffusion model to spread detected disasters easily. However, the system needs well-defined geographic area information to detect and diffuse events. It focuses on the influence diffusion model rather than a local event detection method.

Di Girolamo et al. [31] proposed an online event detection scheme based on a game theoretical clustering. First of all, it is not a local event detection scheme that uses geographic information. However, the authors proposed a general-purpose event detection system that can be applied to various online social networks (OSNs), as well as Twitter. In addition, the system shows a vast amount of big data processing and demonstrates its feasibility by actively utilizing big data platforms such as Cassandra, Kafka, and Spark.

Existing event detection schemes can be classified for various purposes. Real-time event detection schemes [17,22,23,26,28,30,31] assume situations where information that must be provided to users in real time is critical. General event detection schemes [6,13] do not have a definition of events, but they are currently aimed at detecting various issues occurring online. Local event detection schemes [17,22–24,27,28,30] aim to detect events or issues occurring in a particular region in the real world. In certain studies, there are also schemes to detect only predefined events [17,23]. The proposed scheme in our work is not intended for processing in real time, but it is possible to provide events in a specified time unit (1 h). Furthermore, it detects local events through relevant document analysis and a geographic dictionary and does not restrict the definition of events.

There are three main schemes to detect events. First, there is a scheme to detect an event using the burstiness of a particular word through a language analysis of a document [6,22]. It is not suitable for a local event detection scheme because it analyzes only words. The second is a clustering-based scheme that builds a graph according to word frequency and graph generation rules and detects events through clustering. It is easy to analyze associations through graph analysis, but one problem is that the analysis time is not fast. Finally, a learning-based detection scheme is used for detecting events [17,23,24,28,30] through deep learning models such as LSTM or through various learning algorithms. They can improve their accuracy through learning, but there is a disadvantage of having to learn in advance through predetermined events. The proposed scheme detects events based on clustering techniques. To improve the local event detection rate, a relevant document analysis is performed, and the analysis is continuously reflected in the graph to detect the event.

The proposed scheme can provide rich geographic information compared to the existing schemes through the relevant document analysis and a geographic dictionary. However, the geographic dictionary should consist only of words that can accurately distinguish geographical information. The quality of the geographic dictionary affects the local information of the event, so it is necessary to build a reliable geographic dictionary.

3. The Proposed Local Event Detection Scheme

3.1. System Architecture

The existing local event detection schemes use 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, the scheme needs to be complemented with additional geographical data. In or-

der 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 the geo-tag and text mining based schemes. We embed geographical information to the detected events by using text mining techniques and a geographical dictionary if the events 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 a tweet is originally restricted to 140 characters. Sometimes, users need more than one tweet to express their opinions or thoughts. In order to overcome the character restriction of the tweet, Twitter provides a function called a thread, which 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.

Figure 1 shows an overall architecture of the proposed local event detection scheme. The proposed scheme consists of four modules: data collection, graph modeling, relevant analysis, and local event detection. Between each module, the collected SNS data and graph data generated by the proposed scheme are transmitted. The green dotted line represents the transmission of the SNS data, while the red solid line represents the graph data. 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 a local event detection algorithm in sequence through three phases, namely graph modeling, relevant analysis, and local event detection. The first phase, the graph modeling module, constructs keyword graphs through the preprocessed data and the preconstructed 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, and if users do not provide geo-tags, we extract regional keywords through text mining algorithms and a 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. In 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, which we set as 1 h. 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 preprocessed relevant documents, and then it changes the keyword graph constructed from the graph modeling module by using extracted geographical information. In 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 targets only Korean documents. However, this idea can be applied to any language by properly tuning the geographical dictionary and lexical analyzer to suit each language. Buzzwords are relatively common words compared to other words, so they are excluded because they cause noise in event detection. In addition, both emoticons and hashtags can be analyzed, but only the content of the tweet was analyzed in the current paper.

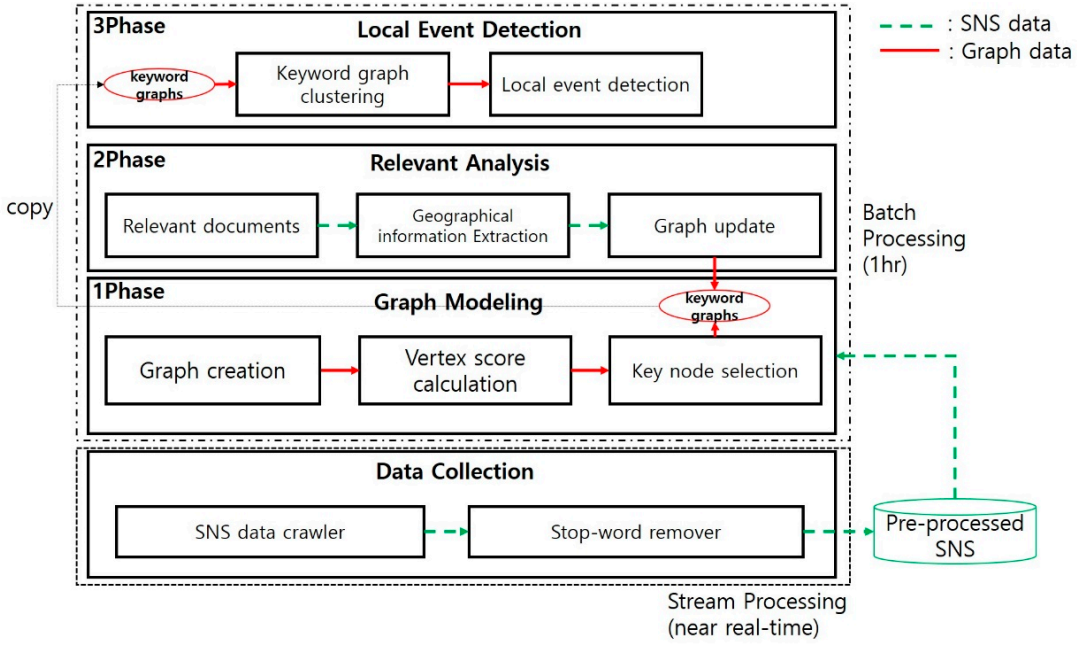


Figure 1. Overall architecture of the proposed local event detection scheme.

3.2. Data Collection

In order to detect local events in SNSs, we should collect SNS data generated by users in real time. 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 in order to extract meaningful keywords.

Figure 2 shows the processing procedure of the data collection. It collects SNS data such as posts (i.e., tweets), comments, retweets, 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 preconstructed. 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 verbs, adjectives, adverbs, prepositions, 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.

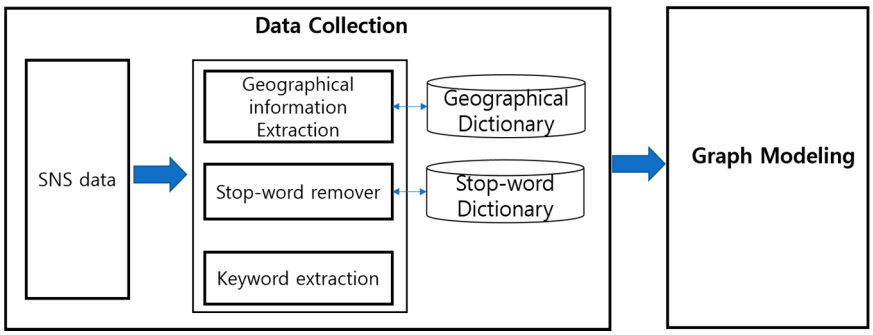


Figure 2. Processing procedure of the data collection.

3.3. Graph Modeling

We gathered keyword sets through the data collection. However, the keyword sets may contain a lot of unnecessary information unrelated to the event. Therefore, we generated a keyword graph that considers word importance and the number of mentions to refine the keyword sets that contain ambiguous information. This allowed us to easily figure out a keyword correlation and importance through the keyword graph, in which vertices and edges are assigned a weight value according to the word importance and the number of mentions. Figure 3 shows the overall procedure of selecting the "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 considered users' explicit opinions in SNSs such as the number of likes and the number of retweets. We used geo-tag information for choosing regional nodes when the keyword graph is constructed. We embedded geographical data using the geographical dictionary if there are no geo-tag data available. We then constructed the keyword graph, where weight values are assigned to vertices and edges based on the number of co-occurrences and the number of mentions. Lastly, we performed an initial graph modeling.

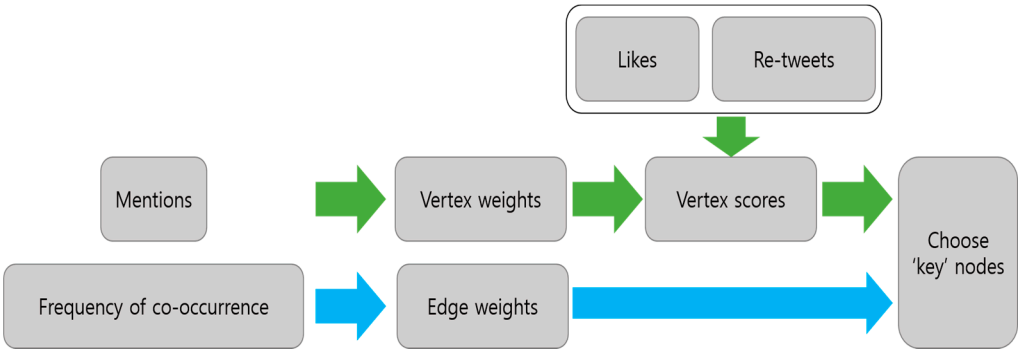


Figure 3. Overall procedure of selecting the key nodes.

We assigned weight values to vertices and edges so that we can 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, retweets, and mentions. The frequency of co-occurrences represents 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 the frequency of co-occurrences for analyzing this situation. The frequency of co-occurrences is changed according to a particular window size that a user has set up. If the window size is 1, the words that come before and after a specific word are included into 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 ew_{ij}, ew_{max}, and ew_{min} denote the frequency of the co-occurrence of vertices i and j, i.e., the highest co-occurrence value and the lowest co-occurrence value, respectively.

$$ew_{ij} = \frac{ew_{ij} - ew_{min}}{ew_{max} - ew_{min}} \quad (1)$$

If we assign the same weight to all vertices that occur within the particular time window t, it is hard to determine which keywords are important and are related to events. Therefore, we consider users' explicit opinions and interests to calculate vertex scores since they 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. The proposed scheme calculates the term frequency (TF) of keyword i, much 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 ($t-1$). Inverse document frequency is a key part of the term frequency–inverse document frequency (TF-IDF) calculation that determines the importance of keywords. The term frequency represents the number of times a term occurs in a document. The inverse document frequency is a measure of how much information the word provides, i.e., whether it is common or rare across all documents. TF-IDF is calculated by the multiplication of TF and IDF. A high weight in TF-IDF is reached by a high term frequency in a given document and a low document frequency of the term in the whole collection of documents; the weights hence tend to filter out common terms.

$$vw_{it} = tf_{it} * \frac{idf_{it}}{idf_{i(t-1)}} \tag{2}$$

We measured vertex scores by using vertex weights and users' explicit opinions, such as like and retweets. 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 l_{it} and r_{it} denote the number of likes of keyword i and the number of retweets of the keyword within the particular time window t, respectively. We applied the logarithm to the sum of likes and retweets for scaling.

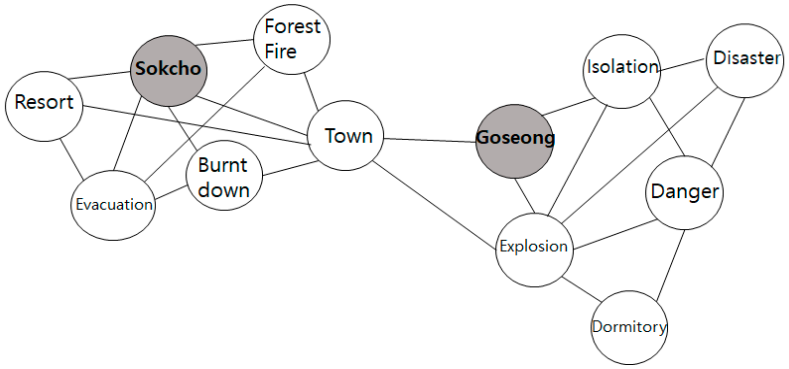
$$vs_{it} = vw_{it} * \log(l_{it} + r_{it}) \tag{3}$$

In order to identify the key nodes, we measured using Text-Rank [32] scores on the weighted keyword graph. Text-Rank is frequently used for extracting primary keywords from the entire graph. We performed clustering based on the key nodes that were selected according to the Text-Rank algorithm. The Text-Rank score can be formulated by Equation (4), where v_{it}, N_i, and d denote a particular vertex i over time window t, neighbors of vertex i in the weighted keyword graph, and the damping factor, respectively. We initialized the $tr(v_{it})$ value with vs_{it}, which is the vertex score calculated by Equation (3). The factor d that adjusts the random probability variable is usually set to 0.85 [33].

$$tr(v_{it}) = (1-d) + d * \sum_{j \in N_i} \frac{ew_{ij}}{\sum_{k \in N_j} ew_{jk}} tr(v_{jt}), \tag{4}$$

The graph clustering based on the key nodes makes it easier to detect words associated with the event and improves the performance compared to using graph clustering based on entire keyword vertices. We chose the 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 were also chosen as key nodes since they have an important role in detecting local events.

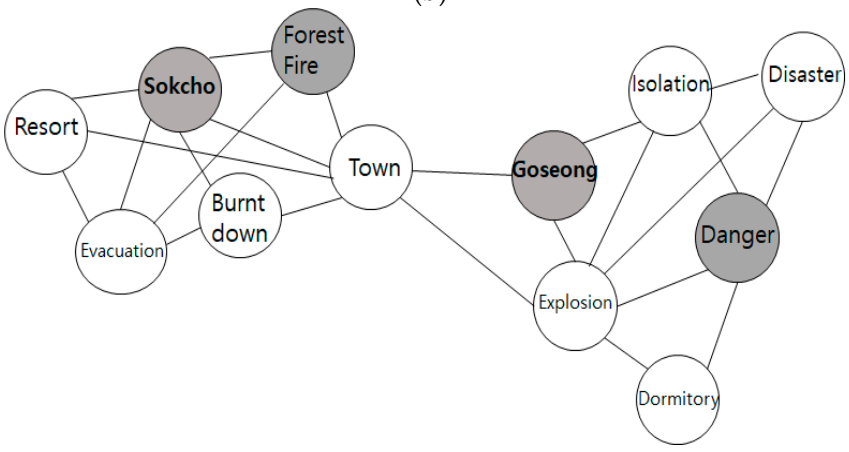
Figure 4 shows the procedure of constructing the keyword graph and choosing key nodes. We constructed the initial keyword graph based on preprocessed data as shown in Figure 4a. The locations "Sokcho" and "Goseong" are labeled as regional nodes. We calculated and assigned vertex and edge weights by using Equations (1) and (2). After that, we performed a Text-Rank analysis for the graph. First, we computed vertex scores by using Equation (3). Second, we assigned a vertex score to each vertex of the original graph. Finally, we determined the Text-Rank scores by using Equation (4). We sorted the Text-Rank scores by descending order and select top-k (k is 4 in this example) nodes (i.e., "Forest Fire", "Sokcho", "Goseong", and "Danger"), as shown in Figure 4b. Finally, we obtained a graph with the key nodes as shown in Figure 4c. A graph is generated based on the tweets that occur within a given unit of time window (1 h). 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 be distinguished among local events.



(**a**)

Rank	Keyword	$tr(v_{i,t})$
1	**Forest Fire**	**31.36**
2	**Sokcho**	**17.67**
3	**Goseong**	**17.10**
4	**Danger**	**15.32**
5	Evacuation	14.84
...		
12	Isolation	9.91

(**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

We need 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. Documents with geographic information, which is very useful for detecting local events, are used especially 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 performed local event detection through an analysis of relevant

documents. We can also easily update the graph because relevant documents already express relationships among postings.

Figure 5 shows an example of relevant documents. Although the tweet of User 1 includes event information, it is difficult to determine a particular region. Region information is added through User 1′s answer to User 2′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.

Algorithm 1 shows the relevant document analysis algorithm. The input parameters are doc (the relevant document) and KG (the keyword graph). We extract keywords (i.e., 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 (Lines 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 (Lines 5–6).

Algorithm 1 AnalysisRelevantDocuments(doc, KG)

```
Input
    doc: Relevant document
    KG: Keyword graph
(n_keys, r_keys) ← extractKeys(doc) // extract keywords from relevant document
if r_keys not empty then
    if r_keys ∈ KG.r_keys or ∀n_keys ∈← KG.n_keys then
        KG.addAndUpdateKeys(n_keys)
    else
        createNewGraph(n_keys, r_keys)
    end if
end if
```

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, which means that the higher the weight is, the more frequently co-occurring the keywords are. The clustering algorithm combines the key nodes with their one-hop neighbors that have a higher edge weight than a threshold (α). The threshold is selected based on the network modularity, which 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 NM_G, m, n, ew_{ij}, and k_i denote the network modularity value of a graph G, 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. In Equation (5), $\delta(c_i, c_j)$ is a boolean function that returns 1 or 0 depending on whether vertex i and vertex j are in the same cluster.

$$NM_G = \frac{1}{2m}\sum_{ij}^{n}[ew_{ij} - \frac{k_i k_j}{2m}]\delta(c_i, c_j) \quad (5)$$

The procedure of merging and splitting clusters is needed for providing "k" events to users, where k is the number of events that we want to get. The proposed scheme provides "k" clusters to users as the local event by using the in-cluster and the out-cluster weight.

Figure 6 shows an example of detecting local events of the proposed scheme. First, we form clusters according to the abovementioned procedure. After that, we obtain 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 C_1 and C_2 are 0.66 and 0.53, respectively. The out-cluster weight between C_1 and C_2 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 C_1 and C_2 since C_1 (or C_2) is closely related with C_2 (or C_1). Likewise, we compare the sum of the in-cluster weights with the out-cluster weight between C_1 and C_3. The out-cluster weight is lower than the sum of the in-cluster weights (in. 0.66 + 0.74 > out. 0.40 + 0.32). Therefore, we remove the connected edges of C_1 and C_3. Finally, we provide users with the detected (clustered) local events as shown in Figure 6b.

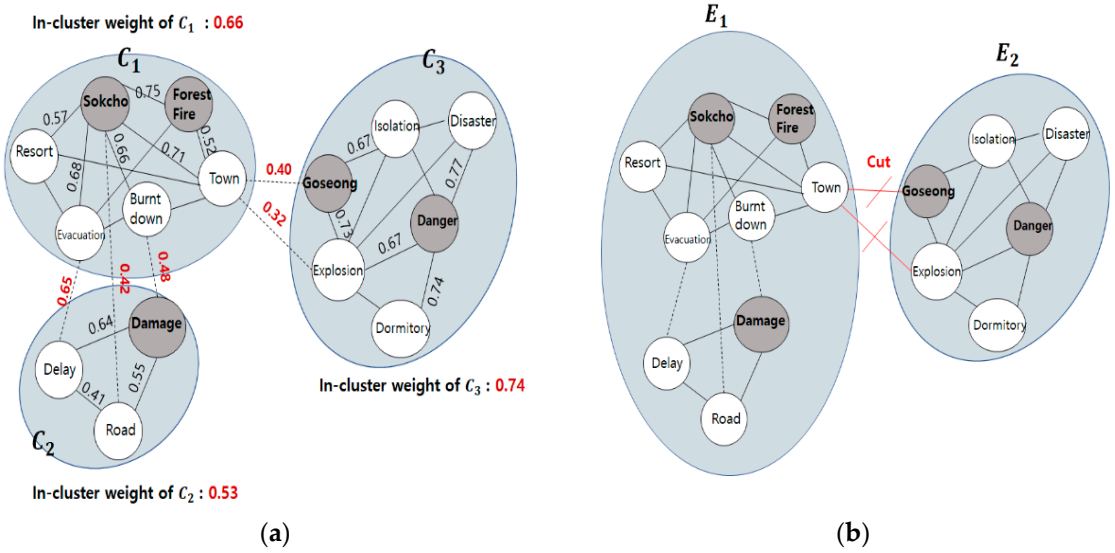


Figure 6. Example of detecting local events: (**a**) clustering; (**b**) detecting local events by merging and splitting clusters.

Algorithm 2 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 the initial values (Lines 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 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 then inserted into the result E (Line 10). We return the result E as a set of the detected local events.

Algorithm 2 DetectLocalEvents(KG, k)

Input
KG: Keyword graph
k: the number of local events
Output
$E_{1..k}$: set of keywords of local events

```
E ← ∅
alpha ← FindBestThreshold(KG, k) // set threshold value using network modularity
CG1...k ← Clustering(KG, k, alpha)
for i = 1, 2, . . . , k − 1 do
    inWeight ← getInClusterWeight(CGi)
    for m = i + 1, i + 2, . . . , k do
        inWeight += getInCluserWeight(CGm)
        outWeightSum ← getOutClusterWeight(CGi, CGm)
        if inWeight < outWeightSum then
            E ← mergeCluster(CGi, CGm)
        end if
    end for
end for
return E
```

4. Experiments

We verified 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 119,243 tweets and 137,942 relevant documents by using Twitter's standard API. We collected 1 month of tweets and documents (from 1 August 2020). The proposed scheme collected only specific topic-based data. In other words, since the aim of the proposed scheme is to detect events occurring in the real world, tweets were collected based on event words such as fires, typhoons, COVID-19, car accidents, earthquakes, floods, and demonstrations that could occur offline. In this paper, the ground truth was annotated by human users. In the experiments, we used 100 local events as the ground truth. The ground truth is called the true set in the paper. In information retrieval, recall is the fraction of the relevant documents that are successfully retrieved [34]. In addition, precision is the fraction of retrieved documents that are relevant to the query on the event. For example, for an event search on a set of documents, precision is the number of correct results divided by the number of all returned results. F-measure is a measure that combines precision and recall; it is the harmonic mean of precision and recall. Therefore, it is calculated as follows:

$$F_{measure} = 2 * \frac{precision * recall}{precision + recall} \quad (6)$$

Table 1. The experimental environment.

Name	Value
Processor	Intel i5-3570k, 3.4 GHz, 4 Core
Memory	16 GB
No. of tweets	119,243
No. of relevant documents	137,942

The recall was calculated by changing k from 5 to 20 (maximum/20 to max/5). Recall calculation values are rounded; k is the value that represents the number of events that each event detection scheme extracts. As we changed k from 5 to 20, we performed a comparison between the search results of the proposed technique and the set of correct answers. Accuracy is calculated by assigning a value of 0.5 if the clustered event is the same as the true set, and 0.5 if the keywords that are considered to have the same meaning as the tagged event category are clustered.

The existing schemes such as text-based event detection [6] and geo-tag based event detection [17] were chosen for comparison. We compared precision, recall, and F-measure to show the excellence of the proposed scheme. The proposed scheme, the text-based event detection [6], and geo-tag based event detection [17] are denoted as T+R, TBEM, and GTBEM, respectively. The denotation OnlyTweet refers to using the event detection scheme without considering relevant documents.

The results of the clustering depend on a threshold. Therefore, we derived an optimized value through experiments. The proposed technique considers one community (cluster) as one event in the graph. NM_G is a quantitative representation of how well each cluster is separated. The higher the value is, the better the separation of events is. Therefore, we find the highest NM_G value while changing the threshold value and utilize the threshold at the highest NM_G for clustering. Figure 7 shows a comparison of NM_G according to different thresholds. In the experiments, thresholds were assigned a value 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 NM_G over the other NM_G when the threshold is 0.5, we set the threshold to 0.5.

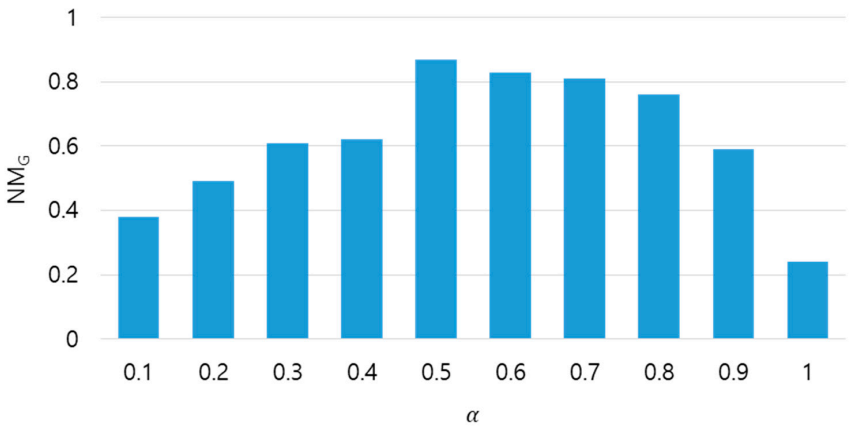


Figure 7. Comparison of NM_G (network modularity) according to different thresholds.

Figure 8 shows a comparison of precision according to the local event detection schemes. GTBEM and TBEM show relatively low performance (about 40%), as it is hard to derive regional information because they use only a text mining algorithm and geo-tags, respectively. GTBEM shows the lowest performance (about 38%) since it only considered geo-tags. OnlyTweet, which does not consider relevant documents, ranked second (about 65%). This means that tweets contained more regional information than geo-tagged tweets. The proposed scheme shows the best performance in terms of precision. The

proposed scheme improved performance by about 20% through the relevant document analysis compared to OnlyTweet.

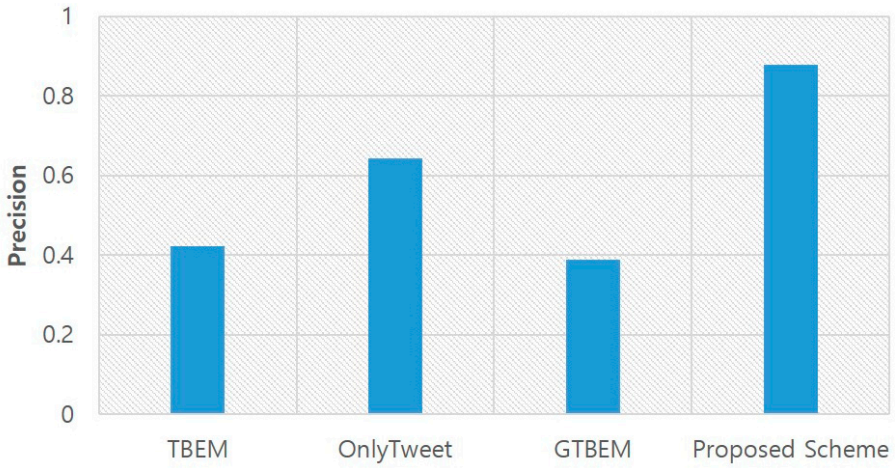


Figure 8. Comparison of precision according to the local event detection schemes.

Figure 9 shows a comparison of recall according to the local event detection schemes. Similar to the experiment results of precision, our scheme achieves the best performance over the existing schemes in improving performance by about 22%. The recall performance of TBEM, OnlyTweet, GTBEM, and the proposed scheme achieved about 40%, 61%, 36%, and 83%, respectively. As shown above, the recall of GTBEM is also much lower than the others, since it is hard to find local events, as it takes only geo-tag information into account, i.e., the detected events with at least 1 or more geo-tag information. The recall performance of TBEM achieved 42% or more, since it is hard to extract regional information. The proposed scheme achieved the highest recall of about 0.83 (83%). This means that relevant local events were well detected, since we utilized not only a geographical dictionary but also relevant documents to complement spare geographical data.

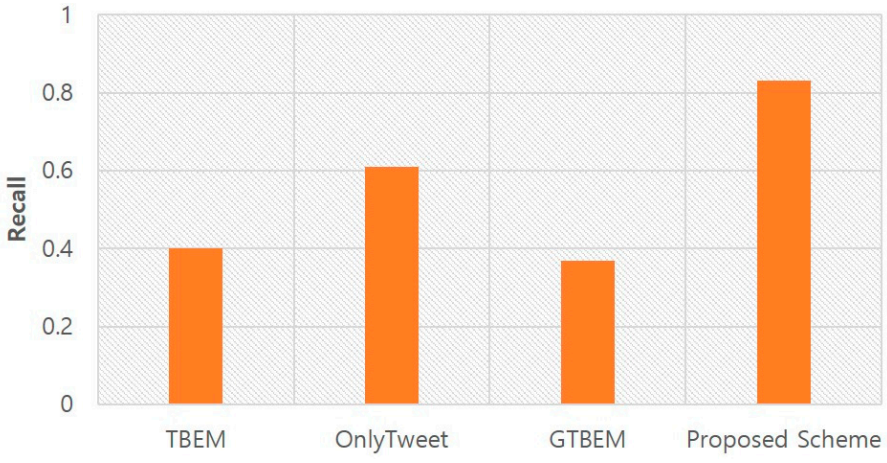


Figure 9. Comparison of recall according to the local event detection schemes.

Figure 10 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, OnlyTweet, GTBEM, and the proposed scheme achieved about 41%, 62%, 37%, and 85%, respectively. As a result, the proposed scheme achieves higher performance than the existing schemes by about 40% or more since it considers both the relevant document analysis based on text mining and geo-tagging.

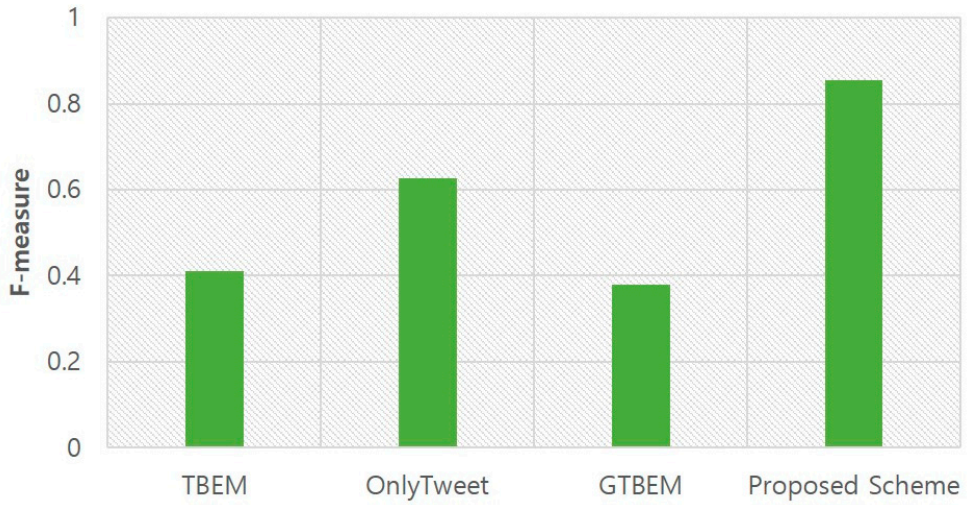


Figure 10. Comparison of F-measure according to the local event detection schemes.

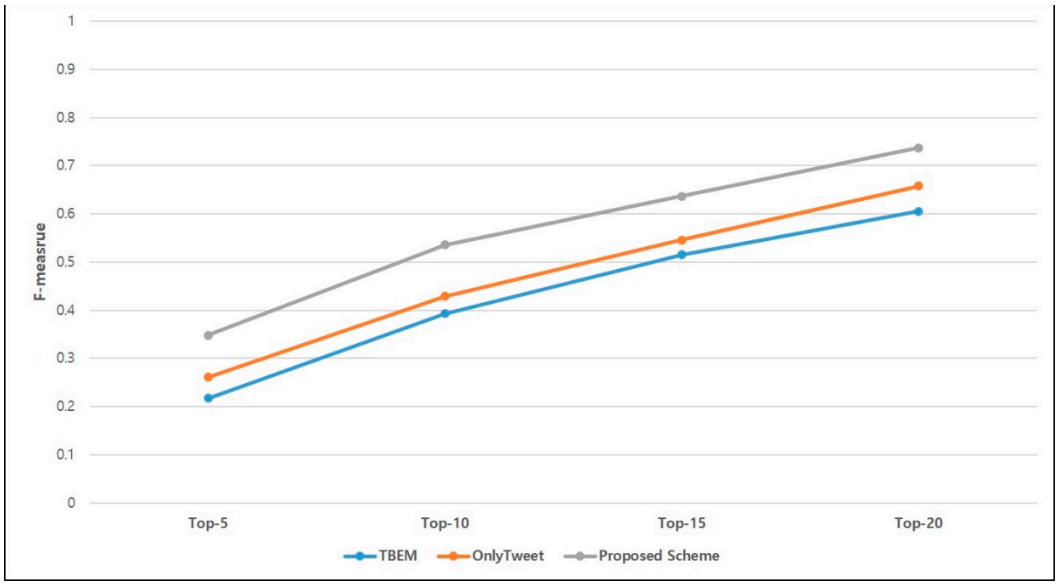


Figure 11. F-measure according to the number of top-k events.

Figure 11 shows the F-measure according to the number of top-k events. When k is 20, all schemes have a high F-measure. The F-measure of TBEM, OnltyTweet, and the proposed scheme achieved about 60%, 65%, and 73%, respectively. The proposed scheme achieves higher performance than the existing schemes by about an average 10% or more when k is 20. The F-measures of all the schemes decrease as k decreases, since recall decreases as the number of detectable events decreases. Given that TBEM cannot detect regional events as it only utilizes text mining techniques, performance results are very low as k decreases. Although TBEM and OnlyTweet can provide the top 20 meaningful events, most of the results did not involve "local" events since it is 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.

5. Discussion

In this section, we present the novelties and limitations of the proposed scheme. First, the novelties of the proposed scheme are as follows.

- Complementing regional information in local event detection: The existing local event detection schemes either use only geo-tagged tweets to extract regional information or use the user's profile and the profiles of the user's friends to infer regional information. However, as mentioned in the existing studies [21,23], geo-tagged tweets account for less than 1% of the total tweets and are even fewer if they only include tweets associated with real events. The proposed scheme complements regional information through a geographical dictionary and the regional information that is referred to in retweets as well as geo-tagged tweets.
- Event detection through relevant document analysis: In Twitter, retweets and threads explicitly indicate an association with the previous tweet. Based on this information, regional information can be supplemented on a prebuilt graph. In addition, information about the event can be expressed in detail on the graph.

Furthermore, the limitations of the proposed scheme derived from experimental evaluations are as follows.

- Difference of detection rate by event: Local event detection schemes show different detection rates depending on the characteristics of the event. For example, COVID-19 frequently generates tweets that contain very close regional information. It shows a very high detection rate through the proposed scheme. Events such as fires and accidents, on the other hand, show relatively low detection rates, as the tweets mainly contain insurance, content related to local fire departments, content related to incidents overseas, and historical content. By reflecting on these characteristics, we need to overcome these limitations in a way that assigns weights to certain words for each event.
- Geographic dictionary dependence: The proposed scheme complements regional information based on a predefined geographic dictionary. However, the proposed scheme has a problem where important events containing specific building names are not frequently detected by the current geographic dictionary. To address such a problem, a novel method is needed to build a learning-based geographic dictionary in which data are constantly added through learning. Furthermore, it is necessary to collect events and geographic information from various cases through more diverse data collection.

6. Conclusions

We proposed an efficient local event detection scheme by analyzing relevant documents in a social network service such as Twitter. The proposed scheme utilizes a geographical dictionary to supplement non-geo-tagged postings in order to detect local events by using geographical information. The proposed scheme 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 the 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 that 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 datasets and conduct additional experiments. In addition, we will 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., D.H., K.B. and J.Y.; validation, D.C., S.P., D.H. 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. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2019R1A2C2084257), by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. B0101-15-0266, Development of High Performance Visual Big Data Discovery Platform for Large-Scale Realtime Data Analysis) Large-Scale Realtime Data Analysis, by the Ministry of Education of the Republic of Korea and the National Research Foundation of Korea (NRF-2017S1A5B8059946), and the MSIT(Ministry of Science and ICT), Korea, under the Grand Information Technology Research Center support program(IITP-2020-0-01462) supervised by the IITP(Institute for Information & communications Technology Planning & Evaluation).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to users' privacy because data include their personal information.

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 Soc.* **2018**, *20*, 162–179. [CrossRef]
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. *J. Assoc. Inf. Sci. Technol.* **2015**, *66*, 2045–2060. [CrossRef]
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. [CrossRef]
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. [CrossRef] [PubMed]
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. [CrossRef]
6. Hossny, A.H.; Mitchell, L. Event Detection in Twitter: A Keyword Volume Approach. In Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW), Singapore, 17–20 November 2018; 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, Maui, HI, USA, 29 October–2 November 2012; 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, New York, NY, USA, 19–23 October 2020; 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. [CrossRef]
10. Gao, W.; Fang, Y.; Wang, Y.; Zhang, F. HRCE: Detecting Food Security Events in Social Media. In *Journal of Physics: Conference Series*; IOP Publishing: Bristol, UK, 2020; p. 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*; Springer: Cham, Switzerland, 2020; 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*; Springer: Singapore, 2020; pp. 229–237.
13. He, J.; Liu, Y.; Jia, Y. EventGraph Based Events Detection in Social Media. In Proceedings of the International Conference of Pioneering Computer Scientists, Engineers and Educators, Zhengzhou, China, 21–23 September 2018; pp. 150–160.
14. Genc, H.; Yilmaz, B. Text-Based Event Detection: Deciphering Date Information using Graph Embeddings. In Proceedings of the International Conference on Big Data Analytics and Knowledge Discovery, Linz, UA, Austria, 26–29 August 2019; pp. 266–278.
15. Liu, Y.; Peng, H.; Li, J.; Song, Y.; Li, X. Event Detection and Evolution in Multi-Lingual Social Streams. *Front. Comput. Sci.* **2020**, *14*, 1–15. [CrossRef]
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*; Springer: Singapore, 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, Halifax, NS, Canada, 13–17 August 2017; 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, Montreal, QC, Canada, 21–26 April 2018; pp. 1–11.

19. Zhang, S.; Cheng, Y.; Ke, D. Event-Radar: Real-Time Local Event Detection System for Geo-Tagged Tweet Streams. *arXiv* **2017**, arXiv:1708.05878.
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 Proceedings of the 2017 IEEE International Conference on Data Mining (ICDM), New Orleans, LA, USA, 18–21 November 2017; pp. 435–444.
21. Abdelhaq, H.; Sengstock, C.; Gertz, M. Eventweet: Online Localized Event Detection from Twitter. *Proc. VLDB Endow.* **2013**, *6*, 1326–1329. [CrossRef]
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, Glasgow, GC, UK, 24–28 October 2011; 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 Trans. Intell. Syst. Technol.* **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, Bellevue, WA, USA, 3–6 November 2015; pp. 1–10.
25. Vanderwende, L.; Suzuki, H.; Brockett, C.; Nenkova, A. Beyond SumBasic: Task-Focused Summarization with Sentence Simplification and Lexical Expansion. *Inf. Process. Manag.* **2007**, *43*, 1606–1618. [CrossRef] [PubMed]
26. Sahin, O.C.; Karagoz, P.; Tatbul, N. Streaming Event Detection in Microblogs: Balancing Accuracy and Performance. In Proceedings of the International Conference on Web Engineering, Daejeon, Korea, 11–14 June 2019; pp. 123–138.
27. Wei, H.; Sankaranarayanan, J.; Samet, H. Enhancing Local Live Tweet Stream to Detect News. *GeoInformatica* **2020**, *24*, 411–441. [CrossRef]
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 Proceedings of the 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 9–12 December 2019; 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, Chicago, IL, USA, 5 November 2019; pp. 1–10.
30. Amato, F.; Moscato, V.; Picariello, A.; Sperlì, G. Extreme Events Management using Multimedia Social Networks. *Future Gener. Comput. Syst.* **2019**, *94*, 444–452. [CrossRef]
31. Di Girolamo, R.; Esposito, C.; Moscato, V.; Sperlí, G. Evolutionary Game Theoretical on-Line Event Detection over Tweet Streams. *Knowl.-Based Syst.* **2020**, *211*, 106563. [CrossRef]
32. Mihalcea, R.; Tarau, P. Textrank: Bringing Order into Text. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain, 25–26 July 2004; pp. 404–411.
33. Brin, S.; Page, L. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Comput. Netw.* **2012**, *58*, 3825–3833. [CrossRef]
34. Allen, K.; Berry, M.M.; Luehrs, F.U., Jr.; Perry, J.W. Machine Literature Searching VIII. Operational Criteria for Designing Information Retrieval Systems. *Am. Doc. (Pre-1986)* **1955**, *6*, 93.