A Web Agent Based on Exploratory Event Mining in Social Media

Norifumi Hirata, Hiroyuki Sano, Robin M. E. Swezey, Shun Shiramatsu, Tadachika Ozono, Toramatsu Shintani
Dept. of Computer Science and Engineering
Graduate School of Engineering, Nagoya Institute of Technology
Nagoya, Aichi, Japan
{*nori, hsano, robin, siramatu, ozono, tora*}@*toralab.org*

***Abstract*—We introduce an exploratory event mining system in social media. A system deals with user's interests by making use of interaction between user input and system output. The system presents related events in the form of graph structures called "event graphs". Event graphs based on users' interests are produced by iterating over event presentation and user selection. The system's user interface helps users obtain a better understanding of the content and background of news events. Our system has two kinds of agents. One is a news agent to detect related news articles and the other is a microblog agent to detect related microblog posts on the Web. A microblog agent detects related microblog posts using news articles that contain news URL references, and using a similarity between news titles and tweets.**

***Keywords*-event mining; news article; microblogging;**

I. Introduction

This paper proposes a Web agent system based on exploratory event mining. Exploratory event mining is based on tracking events with a user interface for making exploratory searches. The exploratory interaction process[1] is information visualization and trial-and-error tactics.

Our goal is to support better user understanding of news events. We focus on two points to achieve this goal. The first is user's interest. User's interest is different each other. The system deals with the problem by exploratory interaction process. The second is other users' opinions. A user can take an event from other users' opinions

In order to identify an event that a user is interested in, a user gives a system a URL of browsing news page. Then a user can get related events and microblog posts. It is helpful to improve understanding of an event. Our system has two kinds of agents. One is a news agent to detect related news articles and the other is a microblog agent to detect related microblog posts. A news agent detects related news and similar events for user interaction. A microblog agent collects microblog posts on the Web from Twitter. A microblog agent detects related tweets using news articles that contain news URL references. Furthermore, a microblog agent detects using a similarity between news titles and tweets.

We previously proposed a system[2] to provide better understanding of news events on the Web by tracking events with a user interface for making exploratory searches. This paper proposes a method to add microblog posts related to an event from social media Twitter.

The remainder of this paper is organized as follows. In Section 2 we show an example of event graphs models to deal with user's interaction. Section 3 explains our system's flow and a method to extract news articles and tweets. In Section 4 we discuss results of related events and tweets. Finally, we conclude the paper with a summary of key points regarding the system.

II. Exploratory Event Mining and Microblog

A. An Event and Event Graph

In this paper, we define an "event" as a set of related articles and tweets. In the Topic Detection and Tracking (TDT)[3], [4] project, an event is a unique occurrence at a point in time. We consider that user's interesting events depend on each other. Related event presentation with no interaction is not sufficient. Using user's interaction can deal with user's preferences.

Event relations are presented as a graph structure in which a graph node is an event and a graph edge is an important word common to the event. Figure 1 shows an example event graph of the system obtaining an article. Graph nodes (e.g., $\mathrm{e}_{\mathrm{i-j}}$) are events and graph edges (e.g. w_{k}) are important words. When the system receives a new article, it presents related events; e_{1-1}, e_{1-2}, and e_{1-3}. Users select the most interesting event (e_{1-1}) from the presented events. Events related to the selected event (e_{1-1}) are presented such as e_{2-1}, e_{2-2}, e_{2-3}, and e_{2-4}. The w_1, w_2, and w_3 on the edges are words important for extracting each event. By repeating event presentation and user selection, a user can receive unique event graphs. Exploratory event mining is to build an event graph by interaction between user input and system output.

Figure 2 shows structures and relations of events, news articles, and microblog posts. Important words relate events. An event relates to some news articles and microblog posts. When a user selects an event, a user can obtain related articles, microblog posts, and similar events using important words.

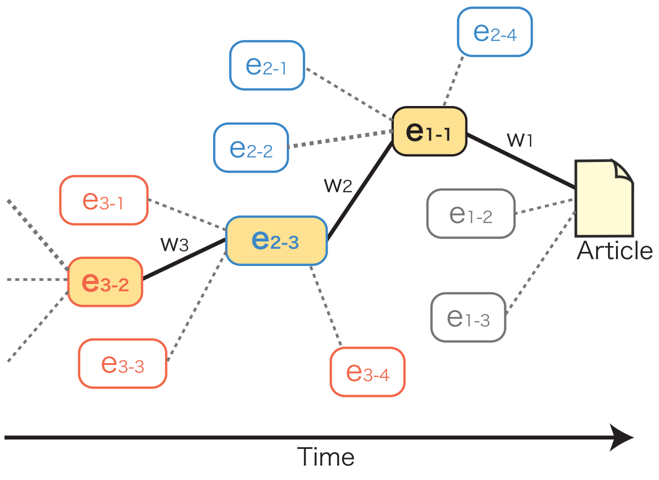

Figure 1. Exploratory event mining and an event graph

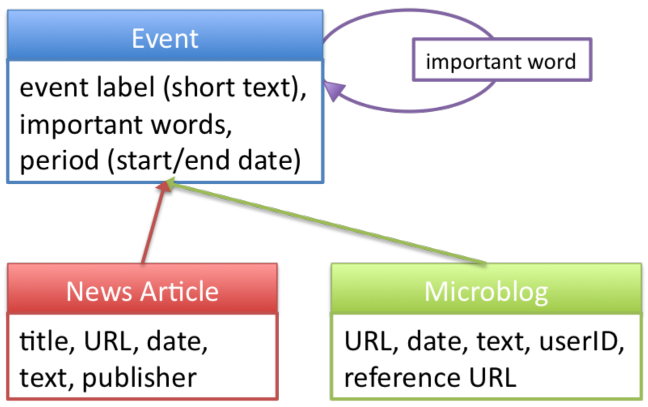

Figure 2. Structure and relation of events, news articles, and microblogs

B. *Related Microblog Posts*

A system can collect related microblog posts using URL of news articles related to an event. When a microblog post refers to URL of a news article, the post is related to the news articles. Some work[5] uses URL links to analyze news and blogs. To collect related posts, other work[6] use hash tags and keyword retrieval. It is well a known method to collect microblog posts about some genres.

It is hard to collect microblog posts related to a news article using only hash tags. Soon after an event occurs, a hash tag to represent the event does not exist. And then microblogging users may make a lot of hash tags for a same event. In some cases, a usage of a hash tag changes. To obtain appropriate hash tags and keywords for each event is not easy. Our system evaluates a similarity between an event and a microblog post. Our system retrieves microblog posts

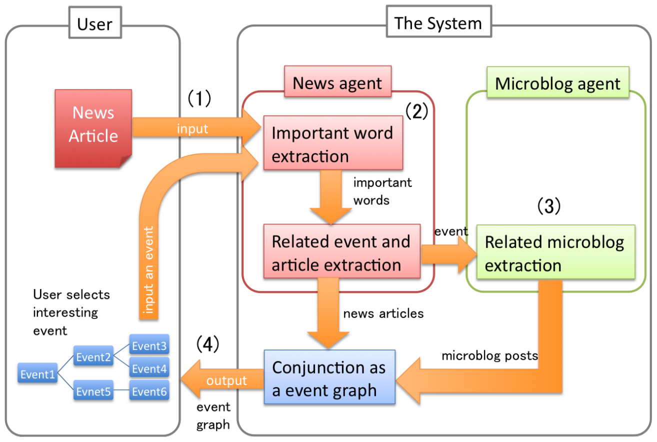

Figure 3. The structure of our event mining system

using keyword conbination based on the similarity.

III. Exploratory Event Mining from News Articles and Microblogs

A. *System Structure*

Figure 3 shows the structure of our event mining system. When a user is interested in a news article, a user can obtain related events, news articles and microblog posts. First, a user inputs a news article to the system. Second, a news agent extracts important words and retrieves related news articles. From related articles the news agent extracts events. Third, a microblog agent extracts tweets related to the events. Finally, the system presents related events. The system outputs events related to the initially input article. A user selects an interesting event from the output events graph. After this, the system input is a selected event. Event graphs are built by iterating over event extraction and user selection.

A news agent crawls and collects news articles from some news sites. A microblog agent collects tweets from the Twitter Streaming API[1]. When a time of an input event is close to a time of system usage, a microblog agent uses the Twitter Search API[2].

B. *Event Extraction using Important Words and Time*

After a user inputs a news article, the first step is to extract important words from it. An important word is a feature word that represents an event. A requirement of an important word is specific parts of speech and high-evaluation. Specific parts of speech are noun and verb. Since words in Japanese sentences are not separated by spaces, the system use a Japanese language morphological analysis program called MeCab [7] for evaluating key speech elements.

[1]https://dev.twitter.com/docs/streaming-api
[2]https://dev.twitter.com/docs/using-search

Evaluation for important words is the sum of the term frequency - inverse document frequency (tf·idf) values of each word w in an event e as shown below:

$$tf \cdot idf_e(w) = \sum_{a \in A_{e_{input}}} tf \cdot idf_a(w) \tag{1}$$

where $A_{e_{input}}$ is a set of news articles related to an input event e_{input}, and $tf \cdot idf_a(w)$ is the term frequency and inverse document frequency value in each article a. When a system input is a news article, an element of $A_{e_{input}}$ is only an input news article. A news agent selects highly evaluated words as important words.

The second step is article retrieval by each important word. To retrieve related news articles, a news agent uses a simple keyword matching and time restriction because a simple method costs few processing time. A news agent selects articles published around the same time because related events occur within similar time.

The final step is related event extraction from retrieved articles. A news agent finds a news article a_{seed} that is the most similar to the input event e_{input}. $t(a)$ means a published time of a news article a. $t(e_{input})$ means an average of $t(a)$ related to a same event. A news agent uses cosine similarity and a time window θ_{ta}.

$$sim_a(e_{input}, a) = \begin{cases} cos(\vec{e_{input}}, \vec{a}), & \text{if } |t(e_{input}) - t(a)| < \theta_{ta} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$$cos(\vec{e_{input}}, \vec{a}) = \frac{\vec{e_{input}} \cdot \vec{a}}{|\vec{e_{input}}||\vec{a}|} \tag{3}$$

$\vec{a}$ is a tf·idf vector of a news article and $\vec{e}$ is a tf·idf vector of an event.

The most similar article a_{seed} is a seed of a cluster.

$$a_{seed} = \max_{a \in A_w} sim_a(e_{input}, a) \tag{4}$$

A_w is a result for article retrieval by important word w.

A news agent adds a news article a to the cluster if the article's similarity exceeds a threshold θ_a. A similarity is cosine similarity between a_{seed} and $a \in A_w$. When a news agent performs for all retrieved articles, the system assumes the cluster as an event.

C. Related Microblog Extraction

When a microblog post refers to URL of a news article, the post is related to the news articles. When a microblog post does not refer to the URL, the system evaluates a similarity between a news article and a microblog post. In this paper, a microblog post is a tweet. The similarity between an article and a tweet is calculated as follows:

$$sim_m(a, m) = \begin{cases} 1, & \text{if } |words(m) \cap url(a)| > 0 \\ f(a, m), & \text{else if} |t(m) - t(a)| < \theta_{tm} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

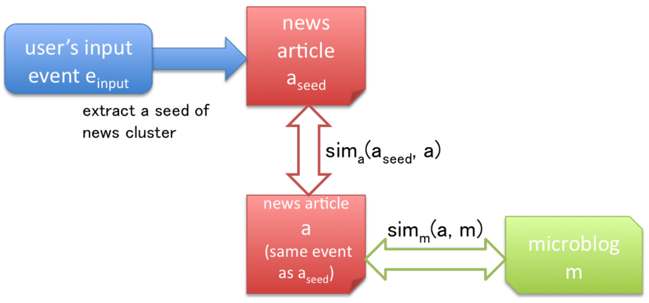
user's input event e_input
news article a_seed
extract a seed of news cluster
sim_a(a_seed, a)
news article a (same event as a_seed)
sim_m(a, m)
microblog m

Figure 4. A relation betweet input event e and tweet m

$$f(a, m) = \frac{\sum_{w \in features(a) \cap words(m)} tf \cdot idf_a(w)}{\sum_{w \in features(a)} tf \cdot idf_a(w)} \tag{6}$$

When a tweet m refers to the URL of news article a, a similarity $sim_m(a, m)$ is 1. $words(m)$ is words in m and $url(a)$ is a URL of a. When a time distance between a tweet and a news article is within a threshold θ_{tm}, a similarity $sim_m(a, m)$ is $f(a, m)$. $t(m)$ is a posted time of m. Time distance is one of important elements because a time and a tweet of a content are closely[9]. In the other case, a similarity $sim_m(a, m)$ is 0. $f(a, m)$ is a rate of news specific words $features(a)$ in tweet text. $features(a)$ is a set of words w that $tf \cdot idf_a(w)$ values exceed a threshold. A similarity between an event e and a tweet m is similar to $sim_m(a, m)$. $sim_m(e,)$

A similarity between an event and a tweet is the maximum value as shown below:

$$sim_{seed}(a_{seed}, m) = \max_{a \in A_e} \{sim_a(a_{seed}, a) \cdot sim_m(a, m)\} \tag{7}$$

a_{seed} is a seed of a news article cluster. A_e is a set of news articles related to event e that is extracted from a_{seed}. If a similarity $sim_m(a_{seed}, m)$ exceeds a threshold θ_m, the system assumes that the tweet m relates to a_{seed}. Figure 4 shows a relation between input event e and tweet m. If a news agent fails to extract a related news article, a microblog agent also fails to extract a related tweet. Thus, a microblog agent calculates sim_{seed} using sim_a and sim_m. When the tweet m is extracted from a news article that has a low similarity sim_a, a condition of a similarity sim_{seed} to exceed a threshold θ_m is hard.

The system cannot evaluate all tweets in the world. Thus the system retrieves tweets using search queries from news articles. The condition that $sim_m(e, m)$ exceeds θ_m is below:

$$S(a) = \left\{ Sub \,\middle|\, Sub \subseteq features(a), \; sim_a(a_{seed}, a) \cdot \frac{\sum_{w_{sub} \in Sub} tf \cdot idf_a(w_{sub})}{\sum_{w \in features(a)} tf \cdot idf_a(w)} \geq \theta_m \right\} \tag{8}$$

```
input: $a_{seed}$ is a seed article for $News$
       $News$ is a set of related news articles
       $Posts$ is a set of tweets
output: a set of related tweets

01: procedure $RetrievePosts(a_{seed}, News, Posts)$
02: begin
03:   $M \leftarrow \{\}$; // a set of related tweets
04:   $Query \leftarrow \{\}$; // a set of queries
05:   foreach $news$ in $News$ do
06:     // $GetQuery$: get $S(news)$
07:     $Query \leftarrow Query \cup GetQuery(a_{seed}, news, \theta_m)$;
08:   end do
09:   // $url(News)$: a set of news URL
10:   $M \leftarrow Retrieve(Posts, url(News))$;
11:   $M \leftarrow M \cup Retrieve(Posts, Query, \theta_{tm})$;
12:   return $M$;
13: end.
```

Figure 5. An algorithm of related tweets extraction using retrieval

Where Sub is a subset of $features(a)$. $tf{\cdot}idf_a(w)$ is a tf·idf value in a news article a. A format of a search query is a disjunctive normal form.

In summary, an algorithm of related tweets extraction is shown in Figure 5. In $GetQuery(a_{seed}, news, \theta_m)$ at line 7, the system calculates a query string according to equation (8). $GetQeury$ does not need tweet data. In $Retrieval(Post, Query, \theta_{tm})$ at line 11, the system retrieves related tweets from Twitter Search API or collected tweets.

D. Labeling of Event

A label of an event is a short text to represent an event. A label is a title selected from news titles because a news title is summary by a newspaper writer. To understand an event content, a news title is easier than a set of words such as tag cloud[8].

We consider that a title that has words of each class is helpful to understand an event. For a user that is unfamiliar with an event, words of some classes such as an actor and a location help to understand. A selected title has oriented information. Proper noun represents location and actor, and verb represents action. Location, actor, and action are important elements to represent an event. The system classifies title words into the five classes that is $C = \{Actor, Location, Action, Other\,proper\,noun, General\,noun\}$. MeCab can classify Japanese words. An event label should have the each element. Evaluation for a title as an event label is a weighted sum of $tf \cdot idf_e(w)$ as shown below:

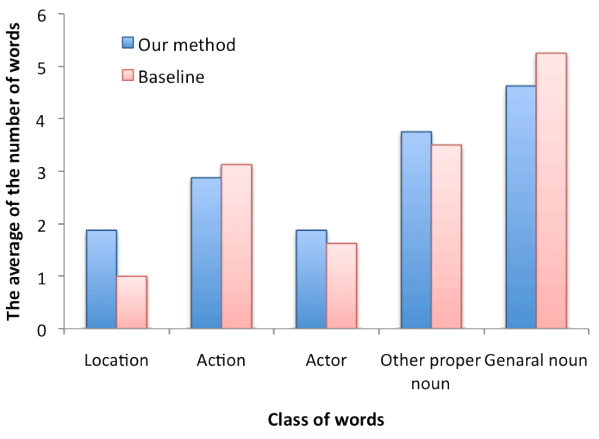

Figure 8. The average number of the words by each class

$$eval(title, e) = \sum_{c \in C} \frac{\sum_{w \in \{w|class(w)=c\} \cap W_{title}} tf \cdot idf_e(w)}{\sum_{w \in \{w|class(w)=c\}} tf \cdot idf_e(w)} \tag{9}$$

where W_{title} is a set of words in a title and $class(w)$ is a class of w.

IV. Evaluation and Examples

A. Example of Event Mining System Usage

Figure 6 and Figure 7 show an example of our system usage. The input article is about Bangkok International Airport resuming its flight schedule after anti-government protesters had ended a blockade of the airport. Rectangle (1) in Figure 6 is a input article. Words in Rectangle (1) are extracted the input article. Rectangle (2) is an extracted event from the important word "Bangkok". Figure 7 is an event graph after an event selection. The system presents events related to the selected event. Exploratory event mining is based on repeating event presentation and user selection.

A user can read related news articles and tweets when a user click a "+" button. Words on edges are important words that are extracted from a selected event.

B. Experiment for Event Label

We compared our method and a baseline method for event labeling. In our method, words are classified to 5 classes. In a baseline method, words are not classified. $eval_{base}(title, e)$ is evaluation for a title as an event label in a baseline method. $eval_{base}$ is a sum of $tf \cdot idf_e(w)$ that w appears in $title$ and $class(w)$ is an element of the classes C.

$$eval_{base}(title, e) = \sum_{w \in \{w|class(w) \in C\} \cap W_{title}} tf \cdot idf_e(w) \tag{10}$$

We used 8 events that has 20.6 news articles on average and compared the selected titles. Figure 8 shows the average

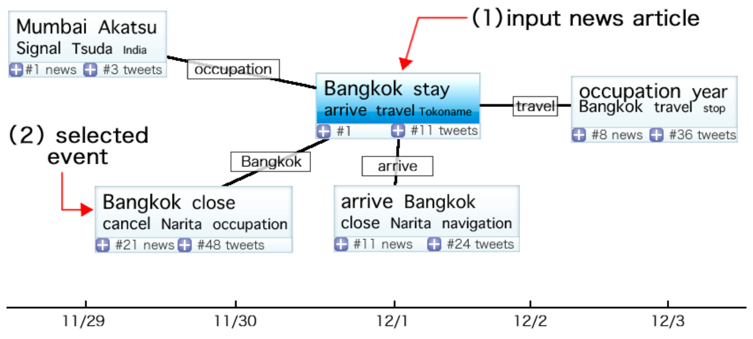
(1)input news article
Mumbai Akatsu Signal Tsuda India
#1 news #3 tweets
occupation
Bangkok stay arrive travel Tokoname
#1 #11 tweets
travel
occupation year Bangkok travel stop
#8 news #36 tweets
(2) selected event
Bangkok
arrive
Bangkok close cancel Narita occupation
#21 news #48 tweets
arrive Bangkok close Narita navigation
#11 news #24 tweets
11/29
11/30
12/1
12/2
12/3

Figure 6. An event graph before an event selection

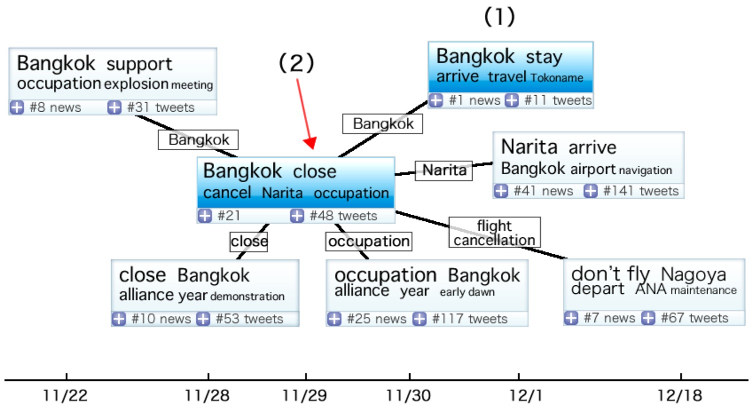
(1)
(2)
Bangkok support occupation explosion meeting
#8 news #31 tweets
Bangkok stay arrive travel Tokoname
#1 news #11 tweets
Bangkok
Bangkok
Bangkok close cancel Narita occupation
#21 #48 tweets
Narita
Narita arrive Bangkok airport navigation
#41 news #141 tweets
close
occupation
flight cancellation
close Bangkok alliance year demonstration
#10 news #53 tweets
occupation Bangkok alliance year early dawn
#25 news #117 tweets
don't fly Nagoya depart ANA maintenance
#7 news #67 tweets
11/22
11/28
11/29
11/30
12/1
12/18

Figure 7. An event graph after an event selection

Table I
THE AVERAGE WORDS AND THE STANDARD DEVIATION IN EVENT LABELS

	our method	baseline
The average number of words in titles	3.00	2.90
The average of standard deviation by classes	1.62	2.10

number of words by each class. The average number of words about location in a title increased from 1.0 to 1.9. Table I shows the average words and the standard deviation in labels. Our method can select a title that has a low standard deviation value. The result shows that a label using our method has a higher probability that contains words of each class. A title that contains a lot of classes is helpful to understand an event at first sight.

C. Experiment for Extracted Microblog Posts in an Event

We compared the result of extracted microblog posts using method A and B. Method A is a method using news titles. Method B is a method that replace a title with a title and a first sentence at $features(a)$ of equation (8). Characteristic words tend to appear in an opening sentence of a new article[10]. Tweets are extracted from the Streaming API (sample). We set that a time windowθ_{ta} and θ_{tm} were 24 hours, a threshold about news articles θ_a was 0.5, a threshold about tweets θ_m was 0.2. A microblog agent extracted related tweets using the 10 seed news articles. The 10 events had 2.2 news articles on average. The 10 events were small-scale events. A small-scale event does not have sufficient related opinions. When an event is small scale, related opinions are necessary.

An average precision is shown as:

$$precision(A_{seed}) = \frac{1}{|A_{seed}|} \sum_{a \in A_{seed}} \frac{|related(a) \cap extracted(a)|}{|extracted(a)|} \quad (11)$$

Where A_{seed} is a set of 10 seed articles. $extracted(a)$ is a result of microblog extraction. $related(a)$ is a set of tweets related to a_{seed}. Table II shows exsamples of related tweets. In this case, a_{seed} is a news article about Osaka Metropolis plan. Type m_A tweets are counted in $|related(a)|$. Type m_B and m_C tweets are not counted.

Table III shows results of microblog extraction. The

Table II
EXSAMPLES OF MICROBLOG POSTS RELATED TO A NEWS ARTICLE

	a part of tweet text	evaluation of relationship
m_A	Sakai city is not up for the Osaka metropolis plan.	relate
m_B	need replacement of fossil fuel: requisition of a regulation to Osaka city	not relate
m_C	contents of conversation: tax payment, annual pension, Osaka Metropolis plan...	hard to evaluate

Table III
RESULTS OF MICROBLOG EXTRACTION

Method	precision	the number of related tweets: $\sum_{a\in A} \lvert related(a) \cap extracted(a)\rvert$	the number of extracted tweets: $\sum_{a\in A} \lvert extracted(a)\rvert$
A	90.1	124	139
B	86.4	127	147
A(72H)	83.6	168	201

results using Method A and B are similar. The precision difference between Method A and Method B is 3.7 point.The results show that a microblog agent obtains sufficient tweets using only titles.

Method A(72H) is based on a Method A. The time window θ_{tm} in a Method A(72H) is 72 hours. A time window θ_{tm} affects a result of microblog extraction from a news article a. The precision using Method A is 89.2 and the precision using Method A(72H) is 83.6.The number of related tweets increased from 124 to 168. A microblog agent can obtains more tweets using a wide time window θ_{tm}. However, a precision tends to decrease.

V. CONCLUSION

We described an exploratory event mining system in social media. Our goal is to support better user understanding of news events. Our system focuses on user's interests and other users' opinions. Our system deals with user's interests by making use of interaction between user input and system output. Microblog posts about an event help users obtain a better understanding. We confirmed that a user obtainded microblog posts using our proposed similarity.

In our related work[11], we classified microblog posts according to regions. Many priorities of presentation can deal with each user's preference. In extraction of news articles and microblog posts, thresholds (θ_a and θ_m) are important for the performance of the system. Higher thresholds make a higher precision and a lower recall. Additional work is required for a priority to present event and related information.

ACKNOWLEDGMENT

This work was supported in part by SCOPE (Strategic Information and Communications R&D Promotion Programme) from Japan's Ministry of Internal Affairs and Communications.

REFERENCES

[1] G. Marchionini, "Exploratory Search, From Finding to Understanding", in Communication of the ACM, Vol. 49, No. 4, pp. 41-46, 2006.

[2] N. Hirata, S. Shiramatsu, T. Ozono and T. Shintani, "Generating an event arrangement for understanding news articles on the web", in Proc. of the 23rd. International Conference on Industrial Engineering and Other Applications of Applied Intelligence Systems, vol.6097, pp. 525–534, 2010.

[3] J. Allan, J. Carbonell, G. Doddington, J. Yamron and Y. Yang, "Topic detection and tracking pilot study final report", in Proc. of the DARPA broadcast news transcription and understanding workshop, pp. 194–218, 1998.

[4] D. Trieschnigg and W. Kraaij, "Tno hierarchical topic detection report at tdt 2004", in Topic Detection and Tracking 2004 Workshop, 2004.

[5] M. Gamon, S, Basu, D. Belenko D. Fisher, M. Hurst and A. C. Knig, "BLEWS: Using Blogs to Provide Context for News Articles", Proc. of the 2nd AAAI Conference on Weblogs and Social Media, 2008.

[6] B. Sharifi, M. A. Hutton and J. Kalita, "Summarizing microblogs automatically", Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 685–688, 2010.

[7] MeCab: Yet Another Part-of-Speech and Morphological Analyzer, http://mecab.sourceforge.net

[8] A. W. Rivadeneira, Gruen, D. M. Gruen, M. J. Muller and D. R. Millen, "Getting our head in the clouds: toward evaluation studies of tagclouds", Proc. of the SIGCHI conference on Human factors in computing systems, pp. 995–998, 2007.

[9] K. Lerman and R. Ghosh, "Information Contagion: an Empirical Study of the Spread of News on Digg and Twitter Social Networks", The 4th International AAAI Conference on Weblogs and Social Media, pp.90–97 2010.

[10] Kyodo News, "Handbook for Editors & Writers 12th edition", 2010.

[11] R. Swezey, S. Shiramatsu, T. Ozono and T. Shintani, "Intelligent Page Recommender Agents: Real-Time Content, Delivery for Articles and Pages Related to Similar Topics", In Proc. of the 24th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, vol.6704, pp.173–182, 2011.