Indexing Evolving Events from Tweet Streams

(Extended Abstract)

Hongyun Cai[1,2], Zi Huang[1], Divesh Srivastava[3] and Qing Zhang[2]
[1]School of ITEE, The University of Queensland, Australia
[2]CSIRO ICT Centre, Australia
[3]AT&T Labs-Research, USA
h.cai2@uq.edu.au, huang@itee.uq.edu.au, divesh@research.att.com, qing.zhang@csiro.au

***Abstract*—Tweet streams provide a variety of real-time information on dynamic social events. Although event detection has been actively studied, most of the existing approaches do not address the issue of efficient event monitoring in the presence of a large number of events detected from continuous tweet streams. In this paper, we capture the dynamics of events using four event operations: creation, absorption, split and merge. We also propose a novel event indexing structure, named Multi-layer Inverted List (MIL), for the acceleration of large-scale event search and update. We thoroughly study the problem of nearest neighbour search using MIL based on upper bound pruning. Extensive experiments have been conducted on a large-scale tweet dataset. The results demonstrate the promising performance of our method in terms of both efficiency and effectiveness.**

I. Introduction

Twitter is one of the most popular online social networking services. The quickly-updated tweets cover a wide variety of dynamically changing events. The evolution of an event unfolds its history, providing opportunities for timely responses at different states. Hence, efficient and effective monitoring of event evolution from tweet streams is of great importance.

Tweets are short and noisy, which poses several challenges to event evolution monitoring. Existing methods, such as SPIC [1], can hardly be directly applied because they do not capture the event evolutions and do not support indexing. Without an index, the performance will deteriorate when the data size grows quickly. However, indexing structures designed for long text documents (e.g., the inverted file) cannot efficiently process short and rapidly arriving tweets. To the best of our knowledge, only one event indexing structure (i.e., VDEH [2]) has been proposed before. Nevertheless, the data in VDEH contains all the tweets in each event, which leads to a large storage cost. Most of the existing event evolution monitoring studies focus on tracking the details of only one event. The relationships among multiple events are only monitored in eTrack [3] based on a time window strategy. Therefore, they fail to monitor event evolutions in real time. Further, the setting of the time window length could be problematic.

In view of the lack of effective methods for monitoring evolving Twitter events, we design four event operations to capture dynamic event evolution patterns. Further, the Multi-layer Inverted List (MIL) is proposed as an event indexing structure to support both efficient event search and real-time event update. The MIL is designed in a way that more relevant yet shorter event lists are quickly found at the lowest layer, hence searching longer event lists at the upper layers can be largely avoided by our proposed pruning strategy.

II. Indexing Evolving Twitter Events

The problem we aim to address in this paper is to effectively and efficiently detect emerging events, capture the changes of existing events and eventually reveal the evolution paths of events over continuous tweet streams. Next, we will first introduce our event evolution monitoring strategies, followed by the proposed event indexing structure MIL [4].

A. Monitoring Evolving Events

We define four event evolution operations: creation, absorption, split and merge, to reflect event changes upon the arrival of new tweets. Given a set of N existing events and a new arriving tweet e, we first find the most similar event to e according to the predefined similarity metric φ and denote it as E_{NN}. Based on the similarity constraint θ on the events, one or more of the four operations are triggered by the arrival of e. Specifically, If $\varphi(e, E_{NN}) < \theta$, a new event is **created**, which consists of the single e. If $\varphi(e, E_{NN}) \geq \theta$, the new tweet e is **absorbed** by E_{NN} if the updated event radius still satisfies the constraint θ. Otherwise, E_{NN} will **split** into two new events by applying the bisecting k-Means clustering. After that, the newly split events need to check whether any of them can be **merged** with any existing nearby events, i.e., if the merged events satisfy the similarity constraint.

B. Event Indexing Structure

One key step in event evolution monitoring is to find the nearest neighbour (E_{NN}) of a new tweet. We propose a novel event indexing structure named Multi-layer Inverted List (MIL), along with an upper bound pruning based search strategy to efficiently process the dynamic tweet streams.

The MIL consists of multiple layers as shown in Fig. 1, where each entry at the m-th layer is an m-term. The terms on each layer are sorted in alphabetical order. Each m-term entry in MIL points to a list of events containing the corresponding m words. For instance, the events $E1$, $E6$, $E12$ and $E21$ in Fig. 1 all contain the words "NSA" and "Obama". Given that emergence of events is usually reflected by the change of words' usage, wavelet analysis is applied to filter the trivial words (with low auto-correlations). All the non-trivial single words form the 1-terms set. The remaining m-terms ($m > 1$) are the m-sized frequent word sets mined by FP-growth. Given a new tweet e, the events from a shorter list stored in a lower layer (larger m) are more likely to be relevant because they share more of the same words. This feature motivates our proposed pruning based search strategy as detailed in Section

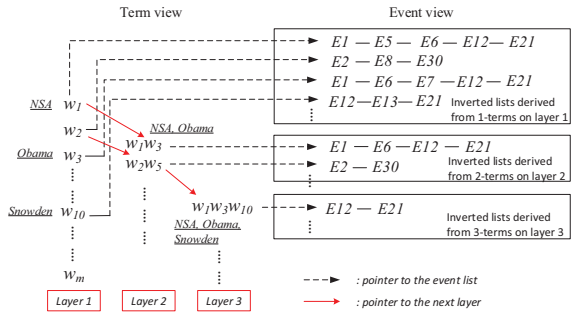


Fig. 1. The Multi-layer Inverted List Structure

II-C. Obviously, the highest layer (i.e., m=1) of MIL alone can be regarded as the conventional inverted file.

C. Event Search

In this work, Cosine similarity is adopted to measure the similarity between an event E and a tweet e, denoted as $\varphi(e, E)$. Given an event E on the m-th layer in MIL, the the upper bound of $\varphi(e, E)$ is estimated as follows:

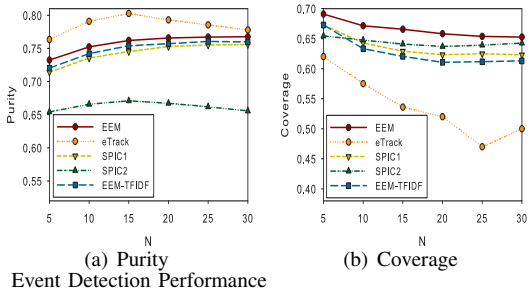
$$\varphi_u^m(e, E) = (\sum_{i=1}^{m} \overline{E_i} + \overline{E_{m+1}} \times (Z - m)) \times \overline{e_1} \quad (1)$$

where $\overline{E_i}$ is the i-th largest value among all dimensions of E, and Z is the number of common words in both E and e.

For each event E on the m-th layer in MIL, a value-pair $< \sum_{i=1}^{m} \overline{E_i}, \overline{E_{m+1}} >$ is stored for upper bound calculation. To find E_{NN} for a new arriving tweet, a depth-first traversal is performed in MIL. Multiple event lists at each layer are combined into a single list. The events belonging to more than one layer are only retained at the lowest layer (largest m) where the tightest upper bounds are computed. Their upper bounds φ_u^m for the query are then computed based on Equation 1. The event list at the lowest level is processed first, followed by its next upper level until the upper most level. The largest similarity value found so far, denoted as $\varphi_{\max}$, is maintained during the process. If the upper bound of an event is greater than $\varphi_{\max}$, the event is accessed and the Cosine similarity is calculated to check whether E_{NN} and $\varphi_{\max}$ need to be updated. Otherwise, it can be safely pruned. Such a pruning based search strategy along with the proposed MIL enable efficient search by first inspecting most relevant event lists on the lowest layer and obtaining a large $\varphi_{\max}$ in short time, to avoid exhaustive accesses to longer event lists at the upper layers. Given that m is generally far smaller than the number of words in tweets (denoted as N), the upper bound computation in Equation 1 (with time complexity $O(1)$) is expected to be more efficient than the Cosine similarity computation (with time complexity $O(N)$). This has been verified in our experiments (Fig. 3).

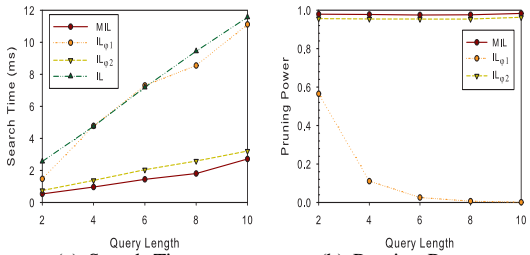
III. Experiments

We collected 11,121,112 tweets posted in 2013 using Twitter API. To evaluate the performance of our proposed event evolution monitoring method (EEM), we compare with a state-of-the-art method eTrack [3] and two variants of SPIC, i.e., SPIC1 with predefined threshold and SPIC2 with auto adjusted threshold. We evaluate the top 30 hottest events for each day and calculate the average purity and coverage across days. As illustrated in Fig. 2, EEM outperforms others in both performance indicators except for purity of eTrack. The high purity of eTrack comes from their proposed skeletal cluster.



(a) Purity (b) Coverage

Fig. 2. Event Detection Performance



(a) Search Time (b) Pruning Power

Fig. 3. Effect of Query Length for Nearest Neighbour Search

However, eTrack tends to recommend similar events, leading to the unsatisfactory coverage value. We further compare the running time between eTrack and our EEM. In terms of the time cost on processing all the tweets with a two-day time window, our proposed EEM outperforms eTrack by reducing more than 70% time cost (i.e., 84,970 ms vs. 285,177 ms).

Fig. 3 reports the effect of query length on the indexing performance for the nearest neighbour search. Clearly, MIL achieves the best search time and highest pruning power. The traditional inverted list indexing (IL) is equipped with two upper bounds, the one proposed in [5] (IL$_{\varphi_1}$) and our proposed bound (IL$_{\varphi_2}$). Our bound significantly outperforms IL$_{\varphi_1}$. By applying the multi-layer structure, the search time and pruning power are further improved. As the query length increases, the superiority of MIL is better demonstrated.

IV. Conclusion

In this paper, we have presented a novel event monitoring method to capture the dynamics of events. A multi-layer event indexing structure is proposed to accelerate the event evolution monitoring process. Extensive experiments are conducted on a real-life tweet dataset to verify the utility of our methods.

Acknowledgment

This work was supported by the Australia Research Council (ARC) under research grant FT130101530.

References

[1] Y. Jie, L. Andrew, C. Mark, R. Bella, and P. Robert, "Using social media to enhance emergency situation awareness," *IEEE Intelligent Systems*, vol. 27, no. 6, pp. 52–59, 2012.

[2] X. Zhou and L. Chen, "Event detection over twitter social media streams," *VLDB J.*, vol. 23, no. 3, pp. 381–400, Jun. 2014.

[3] P. Lee, L. V. S. Lakshmanan, and E. E. Milios, "Incremental cluster evolution tracking from highly dynamic network data," in *ICDE*, 2014, pp. 3–14.

[4] H. Cai, Z. Huang, D. Srivastava, and Q. Zhang, "Indexing evolving events from tweet streams." *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 11, pp. 3001–3015, 2015.

[5] A. C. Awekar and N. F. Samatova, "Fast matching for all pairs similarity search." in *Web Intelligence*, 2009, pp. 295–300.