GENERATION OF ONLINE SUMMARIES IN ARBITRARY TIME DURATION FOR LARGE SCALE TWEET STREAMS

Ramyashree C 1, Mrs. Nandita Bangera2
1 M.Tech student, Department of CSE, AMC Engineering College, Bangalore, India.
2 Assistant Professor, Department of CSE, AMC Engineering College, Bangalore, India

ABSTRACT:

Short-text messages such as tweets are being created and shared at an unprecedented rate. Tweets, in their raw form, while being informative, can also be overwhelming. For both end-users and data analysts, it is a nightmare to plow through millions of tweets which contain enormous amount of noise and redundancy. In this paper, we propose a novel continuous summarization framework called Sumblr to alleviate the problem. In contrast to the traditional document summarization methods which focus on static and small-scale data set, Sumblr is designed to deal with dynamic, fast arriving, and large-scale tweet streams. Our proposed framework consists of three major components. First, we propose an online tweet stream clustering algorithm to cluster tweets and maintain distilled statistics in a data structure called tweet cluster vector (TCV). Second, we develop a TCV-Rank summarization technique for generating online summaries and historical summaries of arbitrary time durations. Third, we design an effective topic evolution detection method, which monitors summary-based/volume-based variations to produce timelines automatically from tweet streams. Our experiments on large-scale real tweets demonstrate the efficiency and effectiveness of our framework.

Keywords: Tweet stream, continuous summarization, summary, timeline.

[1] INTRODUCTION

Popularity of micro blogging services such as Twitter, Weibo, and Tumblr has resulted in the explosion of the amount of short-text messages. For instance, search for a hot topic in Twitter may yield millions of tweets, spanning weeks. To make things worse, new tweets satisfying the filtering criteria may arrive continuously, at an unpredictable rate. One possible solution to information overload problem is summarization. Summarization represents a set of documents by a summary consisting of several sentences. Intuitively, a good summary should cover the main topics (or subtopics) and have diversity among the sentences to reduce redundancy. Unfortunately, existing summarization methods cannot satisfy the requirements because they mainly focus on static and small-sized data sets, and hence are not efficient and scalable for large data sets and data streams. To provide summaries of arbitrary durations, they will have to perform iterative/recursive summarization for possible time duration, which is unacceptable. Their summary results are insensitive to time. Thus it is difficult for them to detect topic evolution. Implementing continuous tweet stream summarization is however not an easy task, since a large number of tweets are meaningless, irrelevant and noisy in nature, due to the social nature of tweeting. Further, tweets are strongly correlated with their posted time and new tweets tend to arrive at a very fast rate.

Let us illustrate the desired properties of a tweet summarization system. Consider a user interested in a topic-related tweet stream, for example, tweets about “Apple”. A tweet summarization system will continuously monitor “Apple” related tweets producing a real
time timeline of the tweet stream. As illustrated in [Figure-1], a user may explore tweets based on a timeline (e.g. Apple” tweets posted from October 22nd, 2012 to November 11th, 2012). Given a timeline range, the summarization system may produce a sequence of time stamped summaries to highlight points where the topic/subtopics evolved in the stream. Such a system will effectively enable the user to learn major news/discussion related to “Apple” without having to read through the entire tweet stream. Given about topic evolution about “Apple”, a user may decide to zoom in to get a more detailed report for a smaller duration (e.g., from 8 am to 11 pm on November 5th). The system may provide a drill-down summary of the duration that enables the user to get additional details for that duration. A user, perusing a drill-down summary, may alternatively zoom out to a coarser range (e.g., October 21st to October 30th) to obtain a roll-up summary of tweets.

![Figure: 1. A Timeline Example of topic “Apple”](image)

In the tweet stream clustering module, we design an efficient tweet stream clustering algorithm, an online algorithm allowing for effective clustering of tweets with only one pass over the data. This employs two data structures to keep important tweet information in clusters. In [Figure-2], the first one is a novel compressed structure called the tweet cluster vector (TCV). TCVs are considered as potential sub-topic delegates and maintained dynamically in memory during stream processing. The second structure is the pyramidal time frame (PTF), which is used to store and organize cluster snapshots at different moments, thus allowing historical tweet data to be retrieved by any arbitrary time durations. The high-level summarization module supports generation of two kinds of summaries: online and historical summaries. To generate online summaries, we propose a TCV-Rank summarization algorithm by referring to the current clusters maintained in memory. This computes centrality scores for tweets kept in TCVs, and selects the top-ranked ones in terms of content coverage.
and novelty. To compute a historical summary where the user specifies an arbitrary time duration, we first retrieve two historical cluster snapshots from the PTF with respect to the two endpoints (the beginning and ending points) of the duration. Then, based on the difference between the two cluster snapshots, the TCV-Rank summarization algorithm is applied to generate summaries. The core of the timeline generation module is a topic evolution detection algorithm, which consumes online/historical summaries to produce real-time/range timelines. The algorithm monitors quantified variation during the course of stream processing. To quantify the summary based variation (SUM), we use the Jensen-Shannon divergence (JSD) to measure the distance between two word distributions in two successive summaries. Second, we monitor the volume-based variation (VOL) which reflects the significance of sub-topic changes, to discover rapid increases (or “spikes”) in the volume of tweets over time. Third, we define the sum-volume variation (SV) by combining both effects of summary content and significance, and detect topic evolution whenever there is a burst in the unified variation. The main contributions of this work are as follows: We propose a continuous tweet stream summarization framework, namely Sumblr, to generate summaries and timelines in the context of streams. We design a novel data structure called TCV for stream processing, and propose the TCV-Rank algorithm for online and historical summarization. We propose a topic evolution detection algorithm which produces timelines by monitoring three kinds of variations. Extensive experiments on real Twitter data sets demonstrate the efficiency and effectiveness of our framework.

![Figure: 2. The framework of Sumblr](image)

### [2] STATE OF THE ART

Stream data clustering has been widely studied in the literature. BIRCH [2] clusters the data based on an in-memory structure called CF-tree instead of the original large dataset. Bradley et al. [3] proposed a scalable clustering framework which selectively stores important portions of the data, and compresses or discards other portions. CluStream [1] is one of the most classic stream clustering methods. It consists of an online micro-clustering component
and an offline macro-clustering component. The pyramidal time frame was also proposed in [1] to recall historical micro clusters for different time durations. A variety of services on the Web such as news filters, text crawling, and topic detecting etc. have posed requirements for text stream clustering. A few algorithms have been proposed to tackle the problem [4], [5], [6], [7]. Most of these techniques adopt partition-based approaches to enable online clustering of stream data. As a consequence, these techniques fail to provide effective analysis on clusters formed over different time durations. Extractive document summarization has received a lot of recent attention. Most of them assign salient scores to sentences of the documents, and select the top-ranked sentences [8]. Some works try to extract summaries without such salient scores. Wang et al. [9] used the symmetric non-negative matrix factorization to cluster sentences and choose sentences in each cluster for summarization.

[3] IMPLEMENTATION

Suppose a tweet t arrives at time ts, and there are N active clusters at that time. The key problem is to decide whether to absorb t into one of the current clusters or upgrade t as a new cluster. We first find the cluster whose centroid is the closest to t specifically, we get the centroid of each cluster compute its cosine similarity to t, and find the cluster Cp with the largest similarity denoted as MaxSim (t). Note that although Cp is the closest to t, it does not mean t naturally belongs to Cp. The reason is that t may still be very distant from Cp. In such case, a new cluster should be created.

The high-level summarization module provides two types of summaries: online and historical summaries. An online summary describes what is currently discussed among the public. Thus, the input for generating online summaries is retrieved directly from the current clusters maintained in memory. On the other hand, a historical summary helps people understand the main happenings during a specific period, which means we need to eliminate the influence of tweet contents from the outside of that period. As a result, retrieval of the required information for generating historical summaries is more complicated.

The core of the timeline generation module is a topic evolution detection algorithm which produces real-time and range timelines in a similar way. We shall only describe the real-time case here. This discovers sub-topic changes by monitoring quantified variations during the course of stream processing. A large variation at a particular moment implies a sub-topic change, which is a new node on the timeline. We first bin the tweets by time (e.g., by day) as the stream proceeds. Though the summary-based variation can reflect sub-topic changes, some of them may not be influential enough. Since many tweets are related to users’ daily life or trivial events, a sub-topic change detected from textual contents may not be significant enough.

[4] RESULT ANALYSIS

PTF has finer granularity of snapshots for more recent moments. As a result, the queried durations can be better approximated. A larger one leads to higher overall quality. Due to larger capacity of each order, PTF with a larger one is able to maintain more snapshots, and thus produce more accurate approximation for the queried durations. Unfortunately, a larger one also requires more storage cost (the numbers in the parentheses
represent the amounts of snapshots in PTF). This is obvious since it enables PTF to store more snapshots, which results in heavier storage burden. For different applications, Sumblr can be customized with different l values. For example, for real-time summarization, a small l is enough; while for historical review, a large l is needed. Granularity. To further evaluate the flexibility of Sumblr, we also conduct a granularity test. We partition the one month data sets into time durations with fixed length (e.g., 24, 72, or 144 hours), then report the average F-scores for these durations under different levels of granularity, summary quality does not have significant difference among different granularities. This is because the quality of a historical summary mainly depends on two endpoints of the duration (i.e., the accuracy of duration approximation in PTF). [Figure-3] gives the following observations: There exists a common trend: more recent time durations have higher summary quality. This is because PTF has finer granularity of snapshots for more recent moments. As a result, the queried durations can be better approximated. A larger l leads to higher overall quality. Due to large capacity of each order, PTF with a larger l is able to maintain more snapshots, and thus produce more accurate approximation for the queried durations. Unfortunately, a larger l also requires more storage cost (the numbers in the parentheses represent the amounts of snapshots in PTF). This is obvious since it enables PTF to store more snapshots, which results in heavier storage burden.

Figure: 3. Quality on Time Duration

[5] CONCLUSION

We proposed a prototype called Sumblr which supported continuous tweet stream summarization. Sumblr employs a tweet stream clustering algorithm to compress tweets into TCVs and maintains them in an online fashion. Then, it uses a TCV-Rank summarization algorithm for generating online summaries and historical summaries with arbitrary time durations. The topic evolution can be detected automatically, allowing Sumblr to produce dynamic timelines for tweet streams. The experimental results demonstrate the efficiency and effectiveness of our method. For future work, we aim to develop a multi-topic version of Sumblr in a distributed system, and evaluate it on more complete and large-scale data sets.
REFERENCES


AUTHORS BRIEF INTRODUCTION

[1] Ramyashree C received BE degree in Computer Science Engineering from Visvesvaraya Technological University of Belgaum in 2014. Currently pursuing M.Tech degree in Visvesvaraya Technological University of Belgaum. Her research interests include Data Mining.

Corresponding Address- #25/5, 1st main, Patel Lakkappa Road, M.S. Layout, Jaraganahalli, J.P.Nagar Post, Bangalore-560078. Phone number-7760763959, 8867825110, 9008490854.

[2] Mrs. Nandita Bangera received M.Tech degree from Visvesvaraya Technological University of Belgaum in 2012. She is currently working as Assistant Professor in AMC Engineering College, Bangalore. Her research interests include Big Data.

Corresponding Address- #25/5, 1st main, Patel Lakkappa Road, M.S. Layout, Jaraganahalli, J.P.Nagar Post, Bangalore-560078. Phone number-7760763959, 8867825110, 9008490854.