Discovering Trending Phrases on Information Streams

Krishna Y. Kamath
Texas A&M University
College Station, TX 77843
kykamath@cs.tamu.edu

James Caverlee
Texas A&M University
College Station, TX 77843
caverlee@cse.tamued.u

ABSTRACT

We study the problem of efficient discovery of trending phrases from high-volume text streams – be they sequences of Twitter messages, email messages, news articles, or other time-stamped text documents. Most exisiting approaches return top-k trending phrases. But, this approach neither guarantees that the top-k phrases returned are all trending, nor that all trending phrases are returned. In addition, the value of k is difficult to set and is indifferent to stream dynamics. Hence, we propose an approach that identifies all the trending phrases in a stream and is flexible to the changing stream properties.

Categories and Subject Descriptors: H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Current awareness systems*

General Terms: Algorithms, Experimentation

Keywords: trending phrases, social media, real-time web

1. INTRODUCTION

High-volume text information streams have grown at an astonishing rate in the past several years. As one example, Twitter has rapidly grown from handling 5,000 tweets per day in 2007 to 50 million tweets per day in 2010 to 140 million per tweets per day in 2011. Making sense of these high-volume text streams offers rich opportunities and there is a growing body of research aimed at mining these streams, e.g., [3, 7, 8, 9]. In this paper, we explore the possibility of extracting trending phrases from these streams for understanding the relationship among phrases, topics, and keywords as they rapidly evolve over time.

2. RELATED WORK

The database community has long studied stream operations, like finding aggregates on time-ordered data streams, say for performing counts, calculating averages, and so forth about stream objects. The *trending phrase discovery* approach at the core of the method described in this paper is similar to the frequent items (FI) finding problem. In the FI problem, a data structure maintains the score of objects seen on the stream, with which the top-k items may be determined. Cormode and Hadjieleftheriou in [3] give details of several algorithms that have been proposed for the FI problem. Based on [3], the solutions to the FI problem can be split into counter-based algorithms [9, 7, 8], quantile algorithms [5, 10] and sketches [2]. Several approaches have been proposed to deal with a variant of the FI problem that decays items over time using a sliding window [4, 1, 6]. We will describe later in the paper how FI problem is different from the problem we are dealing with.

3. TRENDING PHRASES DISCOVERY

In this section, we first describe some concepts and then formally define the trending phrases discovery problem. We then propose our solution to this problem using trend threshold parameters.

Definition 3.1. ***(Information Stream)*** *An information stream $D = \{(d_1, t_1), (d_2, t_1), (d_3, t_2) \dots\}$ is an ordered set of document tuples. The stream D is of infinite size and is ordered in a non-decreasing fashion by time-stamp values of the documents. Note that, there might be multiple documents in the stream that share the same time-stamp.*

Examples of streams that satisfy this property are high volume sequence of Twitter messages, email messages, news articles, or other time-stamped text documents. Our ultimate goal is to map from a stream to a time-sensitive concept hierarchy reflecting the relationship among concepts for a particular degree of temporal granularity (e.g., by minute, hour, or day):

Definition 3.2. ***(Candidate Phrase)*** *Consider a document $d \in D$, where $|d|$ equal to the number of terms in the document. Let P_k be a set of phrases of length k. Then, d can be represented using the set of phrases $P = P_1 \cup P_2 \cup \dots \cup P_{|d|}$. Such a phrase $p \in P$ is called a candidate phrase and the set of all the candidate phrases, observed in the information stream up until time t, is denoted C_t.*

For example, the document "Packers win Superbowl" can be represented by P = {"packers", "win", "superbowl", "packers win", "win superbowl", "packers win superbowl"}. The phrases "superbowl", "packers win", etc are examples of candidate phrases and $P \subseteq C$. A single candidate phrase p may appear multiple times in the stream. Each such instance of the phrase p_i is called a *candidate phrase instance.* For example, consider the document tuples ("Packers win", t_1) and ("Champions packers ", t_2). In this case, the candidate phrase "packers" has two instances each with time-stamp t_1 and t_2 respectively. Similarly, "win" and "champions" have single instances.

PROBLEM 1. ***(Trending Phrases Discovery Problem)*** *Given an infinite information stream D and the set of candidate phrases C_t observed in D, the trending phrases discovery problem is to identify the subset of phrases $\Gamma_t \subseteq C_t$ that are trending at time t.*

One straightforward solution approach is to map this problem to the frequent items finding (FI) problem, where the item score decay with time, and select top-k items as trending phrases. But this approach neither guarantees that the top-k phrases returned are all trending, nor that all trending phrases are returned. In addition, the value of k is difficult to set and is indifferent to stream dynamics. Hence, we propose an approach that identifies all the trending phrases in a stream and is flexible to the changing stream properties.

3.1 Our Solution

The approach we take to identify trending phrases is to associate with each candidate phrase p a score at a particular time $S(p,t)$ and to keep only phrases with a score greater than some threshold, where the threshold may vary in time (as the stream becomes more or less bursty, or as more phrases become "activated"). The challenge is to compute this score efficiently (ideally O(1) time).

Formally, every candidate phrase instance p_i is associated with a score, that decreases as the phrase ages, and a time-stamp t_i of the document in which the instance was observed. The score $S(p_i,t)$ for p_i at time t is given by:

$$S(p_i,t) = \lambda_c^{t-t_i} \qquad (1)$$

where $\lambda_c \in (0,1)$ is a constant know as the *phrase score decay rate*. In a stream we can observe several instances of a candidate phrase p. Let $E(p,t)$ be the set of all the instances of p that have been observed until time t. Then, the *candidate phrase score* at t, $S(p,t)$ is defined as the sum of all the candidate phrase instances of P.

$$S(p,t) = \sum_{p_i \in E(p,t)} S(p_i,t) = \sum_{p_i \in E(p,t)} \lambda_c^{t-t_i}$$

Unfortunately, this definition of $S(p,t)$ requires all of the previous observed instances $E(p,t)$ for p and its calculation takes O($|E(p,t)|$) time, which is inefficient for high-volume text information streams with a large number of candidate phrases. However, we can prove a proposition that will help us calculate this score efficiently in O(1) time and does not have the requirement of storing $E(p,t)$. The proof of the proposition is dropped because of space constraints.

PROPOSITION 3.1. *Given a candidate phrase p with score $S(p,t_l)$ at time t_l, if a phrase instance p_i for p is observed at time t_n, where $t_n > t_l$ then the new score for p at t_n, $S(p,t_n)$ is given as:*

$$S(p,t_n) = \lambda_c^{t_n-t_l} S(p,t_l) + 1$$

Given the scores for all candidate phrases, we can easily determine if a candidate phrase p is a trending phrase if its score is greater than a *trending threshold* θ_{tt}, which yields the set of all trending phrases Γ_t:

$$\Gamma_t = \{p \mid S(p,t) \geq \theta_{tt} \forall p \in C_t\} \qquad (2)$$

Note that the value of θ_{tt} is dependent on the number of candidate phrases in C_t and so can vary with the temporal changes in the information stream.

For example, Figure 1(a) shows the scores for two candidate phrases "super bowl" and "Grammys" over time. To keep the example simple we assume that the phrase rate in the information stream does not change and hence the value of θ_{tt} is unchanged. So, using (2) we have, Γ_{t_1} = {"super bowl"}, Γ_{t_2} = {"super bowl", "Grammys"} and Γ_{t_3} = {"Grammys'}.

Pruning Candidate Phrases: In real-world applications, the number of candidate phrases C_t can become very large, meaning that finding Γ using (2) will become expensive. Hence, we can further prune the set of candidate phrases using a *pruning threshold* θ_{pt} , where $C'_t \subseteq C_t$. Like the trending threshold, the pruning threshold is also dependent on C_t. Using the candidate phrase score, the two thresholds – θ_{tt} and θ_{pt} – divide the candidate phrases C_t into 3 disjoint sets: (i) trending; (ii) transitional; and (iii) non-trending. An example of this is shown in Figure 1(a). Though the candidate phrases to prune are in the non-trending set, not all the phrases in this set can be pruned since the candidate phrases that are in the non-trending set can either be trending up or trending down. Figure 1(a) shows both the cases for the phrase "Super bowl".

The phrases to be pruned are the ones that are trending down, but we cannot determine the direction of the trend just based on the candidate phrase score. Hence, we adopt a heuristic to prune the candidate phrases. For a candidate phrase $p \in C_t$ that is in the non-trending set we use the last time a phrase instance of p was observed, t_p to decide if it should be pruned. To determine C'_t, we propose two approaches that use t_p: (i) a deterministic approach; and (ii) a randomized approach. Both approaches also use a parameter *maximum phrase inactivity time* T_{pi}.

Using the deterministic approach C' is defined as follows:

$$C'_t = \{p \mid S(p,t) \leq \theta_{pt} \textit{ and } (t-t_p) \geq T_{pi} \; \forall p \in C_t\} \qquad (3)$$

The randomized approach uses a function $R(p,t)$ $\forall p \in C_t$ defined as:

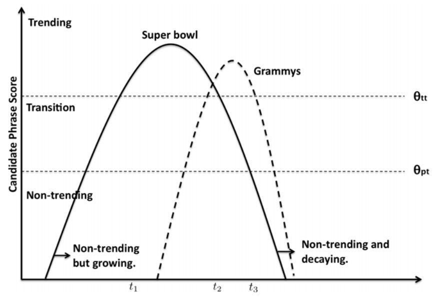
$$R(p,t) = \begin{cases} 0 & \text{if } t - t_p < T_{pi} \\ 1 & \text{if } t - t_p \geq 2T_{pi} \\ coinFlip(\frac{t-t_p}{2T_{pi}}) & \text{Otherwise} \end{cases}$$

where, $coinFlip : [0,1] \rightarrow \{0,1\}$, is a function where the probability of 1 increases as input is closer to 1. Now, C'_t is defined as follows:

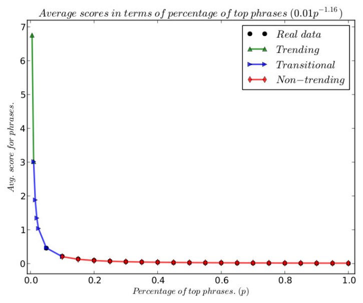
$$C'_t = \{p \mid S(p,t) \leq \theta_{pt} \textit{ and } R(p,t) = 1 \; \forall p \in C_t\} \qquad (4)$$

3.2 Trend Threshold Parameters

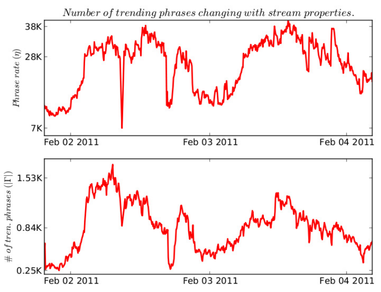
In the previous section we showed how we can efficiently identify trending phrases given some stream-dependent threshold parameters θ_{tt} and θ_{pt}. But how are these parameters determined in the first place? A poor selection can impact the quality and efficiency of concept extraction. A high value



(a) Examples for trending phrases discovery problem.



(b) Phrase scores distribution



(c) Tren. phrases changing with stream

Figure 1

of θ_{tt} may result in false negatives (i.e., incorrectly excluding from concept hierarchy generation a legitimately important concept), while a low value for θ_{tt} may result in false positives (i.e., incorrectly including non-trending phrase concepts). Similarly, a low value of θ_{pt} can result in a large set of candidate phrases C_t making the determination of Γ expensive, while a large value of θ_{pt} results in removal of phrases that could trend in the future, thereby reducing the quality of the results.

Hence, it is important that these trend threshold parameters by determined carefully. Concretely, we first state a lemma that gives us the upper bound on the sum of scores for all the candidate phrase instances that have been observed on the information stream. We will state a theorem, which can be proved using the lemma, that guides the determination of thresholds. The proof of the lemma and theorem are dropped because of space constraints.

LEMMA 3.1. *Given an information stream D with a phrase rate of η and the set of all candidate phrase instances P_t that have been observed on D during time intervals $0, 1, ...t$, we have:*

$$\sum_{p_i \in P_t} S(p_i, t) \leq \frac{\eta}{1-\lambda_c} \quad \text{and} \quad \lim_{t\to\infty} \sum_{p_i \in P_t} S(p_i, t) = \frac{\eta}{1-\lambda_c}$$

THEOREM 3.1. *Given an information stream D at a phrase rate η and the set of candidate phrases C_t at time t, the trending threshold θ_{tt} and the pruning threshold θ_{pt} are given as:*

$$\theta_{tt} = \frac{\eta S_t}{|C_t|(1-\lambda_c)} \qquad \text{and} \qquad \theta_{pt} = \frac{\eta S_p}{|C_t|(1-\lambda_c)}$$

where, $S_t > 1$ and $S_p < S_t$ are stream specific constants.

To summarize, this section has shown how we can efficiently identify the significant base conceptual units (phrases) from an information stream. The two critical trend threshold parameters guiding this identification can be calculated with little overhead and can be tailored for stream and application-specific properties.

4. EXPERIMENTS

In this experiments section, our goal is to see if we can discover trending phrases over a real high-volume text information stream. We want to see if we can estimate the stream parameters required to do such discovery and analyze the properties of the trending phrases discovered.

We begin by examining the algorithm presented in Section 3 to extract trending phrases. Recall that the algorithm requires threshold parameters θ_{tt} and θ_{pt} to determine which phrases to extract, and that these parameters are dependent on stream-specific constants S_t and S_p, respectively. We use a sample of 3.4 million Twitter messages collected from the Twitter Streaming API, resulting in 640 messages per minute.

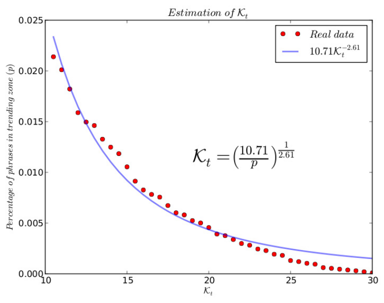
Phrase Score Distribution: First, we analyze the distribution of phrase scores in a sample from the trending topic stream, updating the phrase scores using equation Proposition 3.1. How are these scores distributed? Our goal is to identify a reasonable expected split for trending, transitional, and non-trending phrases (Recall Figure 1(a)).

We set the phrase score decay rate $\lambda_c = 0.80$ and process the stream for 2.5 hours (150 minutes), noting down the phrase scores in C_t every 5 minutes. We then take these 30 sets of phrase scores and sort all of them to calculate the average phrase scores at different levels of top phrases as shown in Figure 1(b). Based on this figure we see that the scores follow a skewed distribution, with most phrases having very low scores (and hence, being of less importance at a particular time in the stream). Based on this score distribution, we fix the following split: (i) Trending (green): top-0.5% phrases (ii) Transitional (blue): middle $0.5 - 10\%$ phrases (iii) Non-trending (red): bottom 90%.[1]

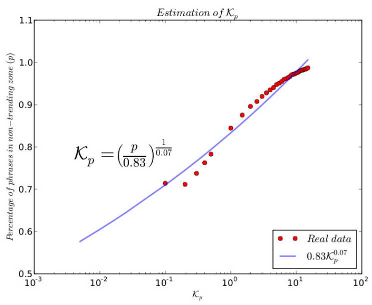
Parameter Estimation: With these splits in mind, we next estimate the stream-specific constants S_t and S_p which guide the threshold parameters θ_{tt} and θ_{pt} for trending phrase discovery as the stream rate (and burstiness) changes over time. To estimate the value of S_t we process the messages from the trending topic stream, and for every 5 minutes interval, we calculate the percentage of phrases above θ_{tt}, calculated using Theorem 3.1. We calculate θ_{tt} for different values of S_t. The values for η and C_t are obtained from the stream, and we set the phrase score decay rate $\lambda_c = 0.80$. The variation of percentage of trending phrases above θ_{tt} with S_t is shown in Figure 2(a). Using this curve we get S_t as a function of the percentage of trending phrases p.

$$S_t = \left(\frac{10.71}{p}\right)^{\frac{1}{2.61}}$$

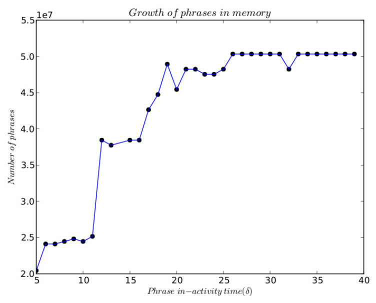
[1]Note that the choice of split is application-dependent. For clarity of presentation in the rest of the paper we select these guideposts for trending phrase discovery.



(a) Estimation of S_t



(b) Estimation of S_p



(c) Estimation of T_{pi}

Figure 2: Parameter estimation.

Hence, if our goal is to consider the top-0.5% of phrases as trending phrases, then for $p = 0.005$ we have the stream-specific constant $S_t = 18.96$.

Similar to S_t, to can calculate S_p over the stream by calculating θ_{pt} for different values of S_p (according to Theorem 3.1). At every 5 minutes interval, we calculate the percentage of phrases that are below θ_{pt}. The variation of percentage of top phrases below θ_{pt} with S_p is shown in Figure 2(b). Using this curve we get S_p as a function of percentage of trending phrases p.

$$S_p = \left(\frac{p}{0.83}\right)^{\frac{1}{0.07}}$$

Hence, for a choice of 90% phrases as non-trending, we have for $p = 0.9$ a score of $S_p = 3.02$.

The values we determined for S_t and S_p are interesting because, using the definitions for threshold parameters θ_{tt} and θ_{pt}, defined in Theorem 3.1, we realize that phrases that have a score more than 19 times the average score are trending while phrases that have score lesser than 3 times the average score can be pruned.

Finally, recall that the set of candidate phrases C_t can be pruned occasionally to remove inactive phrases. The last parameter to be estimated is the maximum phrase inactivity time T_{pi} which guides this pruning. If T_{pi} is low, then the phrases are not allowed to trend, while if it is very high C_t could become too large. Hence, we show in Figure 2(c) the variation of the size of C_t with increasing T_{pi}. We see that the choice of T_{pi} results in a quasi-stepwise change in C_t, and so for this case a choice of $T_{pi} = 7$ minutes would result in a smaller C_t with confidence that the choice is fairly stable.

Stream Properties Vs Trending Phrases: To verify the robustness of our approach to changing stream properties, we observe the number of trending phrases identified by our approach as the input stream changes. This comparison is shown in Figure 1(c). The top plot in the figure shows variation of phrase rate in the input stream for 2 days. We observe that the number of trending phrases identified for concept hierarchy detection (shown in the bottom plot of the figure) correlates with top plot, since the trending threshold θ_{tt} is defined as a function of the phrase rate η.

Sample Trending Phrases: A sample of trending phrases discovered during the first week of Feb, 2011 is shown in Table 1.

5. CONCLUSION

We have studied the problem of efficient discovery of trending phrases from high volume text information streams. We have seen how the proposed method can efficiently identify significant phrases from high-volume information streams. For our future work we are interested in developing applications that use the solution proposed in this paper to discover trending phrases. In this table, we observe various events of the week like Middle East revolution, Super Bowl, transfer of Fernando Torres to Chelsea from Liverpool, etc.

Day	**Trends**
2011-02-01	fernando torres, jan25, human rights
2011-02-02	mubarak, tcyasi, news
2011-02-03	cairo, news, square
2011-02-04	imgood, search google, super bowl
2011-02-05	search google, cowboys, bielsa
2011-02-06	church lady, femme fatale, jon jones
2011-02-06	sb45, chelsea liverpool, lil wayne
2011-02-07	christina aguilera, tomlin, darth vader

Table 1: Sample trend for first week of Feb, 2011

6. REFERENCES

[1] A. Arasu. Approximate counts and quantiles over sliding windows. In *In PODS*, pages 286–296, 2004.

[2] M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. In *Proceedings of International Colloquium on Automata, Languages and Programming*, ICALP '02. Springer-Verlag, 2002.

[3] G. Cormode and M. Hadjieleftheriou. Methods for finding frequent items in data streams. *The VLDB Journal*, 19:3–20, February 2010.

[4] M. Datar, A. Gionis, P. Indyk, and R. Motwani. Maintaining stream statistics over sliding windows (extended abstract), 2002.

[5] M. Greenwald and S. Khanna. Space-efficient online computation of quantile summaries. In *In SIGMOD*, pages 58–66, 2001.

[6] L. K. Lee and H. F. Ting. A simpler and more efficient deterministic scheme for finding frequent items over sliding windows. PODS '06, pages 290–297, 2006.

[7] G. S. Manku and R. Motwani. Approximate frequency counts over data streams, 2002.

[8] A. Metwally, D. Agrawal, and A. E. Abbadi. Efficient computation of frequent and top-k elements in data streams. In *IN ICDT*, pages 398–412, 2005.

[9] J. Misra and D. Gries. Finding repeated elements. Technical report, Ithaca, NY, USA, 1982.

[10] N. Shrivastava, C. Buragohain, D. Agrawal, and S. Suri. Medians and beyond: New aggregation techniques for sensor networks. pages 239–249. ACM Press, 2004.